Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Навчально-науковий центр заочної форми навчання

(повна назва)

Кафедра Інформаційних управляючих систем

(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Розробка сервісу «Відбір пропозицій на розробку ІТ-проектів» інформаційної системи ІТ-компанії

(тема)

Виконав:

здобувач 4 року навчання,

групи ІТУз-21-1

Ілона МАЛАНІЯ

(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки

(код і повна назва спеціальності)

Тип програми освітньо-професійна

(освітньо-професійна або освітньо-наукова)

Освітня програма Інформаційні технології управління

(повна назва освітньої програми)

Керівник: ас. Ірина МАЛЬКОВА

(посада, власне ім'я, прізвище)

Допускається до захисту

Зав. кафедри ІУС _____ Костянтин ПЕТРОВ

(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Навчально-науковий центр заочної форми навчання

Кафедра Інформаційних управляючих систем

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Інформаційні технології управління
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

" 19 " травня 2025 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Маланії Ілоні Давідівні
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка сервісу «Відбір пропозицій на розробку IT-проектів» інформаційної системи IT-компанії

затверджена наказом по університету від " 19 " травня 2025 р. № 82Стз

2. Термін подання здобувачем роботи до екзаменаційної комісії " 12 " червня 2025 р.

3. Вихідні дані до роботи Опис об'єкта автоматизації, схема організаційної структури, структури, альбом документів, що затверджені.

4. Перелік питань, що потрібно опрацювати у роботі Огляд і аналіз сучасного стану розглянутої проблеми, а також існуючих методів і засобів вирішення задач кваліфікаційної роботи, змістовний опис та аналіз структурних і функціональних особливостей об'єкта дослідження та основних видів його забезпечення, постановка задачі кваліфікаційної роботи, обґрунтування мети вирішення поставленої задачі і критеріїв ефективності, розробка і обґрунтування елементів інформаційної забезпечуючої системи; вибір, розробка й обґрунтування елементів програмної та технічної забезпечуючих систем

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи проекту	Терміни виконання етапів роботи	Примітка
1	Змістовний опис та аналіз структурних і функціональних особливостей об'єкта дослідження та основних видів його забезпечення	19.05.2025 – 22.05.2025	Виконано
2	Огляд і аналіз сучасного стану розглянутої задачі, а також існуючих методів і засобів вирішення задач кваліфікаційної роботи	22.05.2025-25.05.2025	Виконано
3	Постановка задачі кваліфікаційної роботи	25.05.2025-26.05.2025	Виконано
4	Обґрунтування мети вирішення поставленої задачі і критеріїв ефективності	26.05.2025-28.05.2025	Виконано
5	Розробка й обґрунтування елементів інформаційної забезпечуючої системи	28.05.2025-02.06.2025	Виконано
6	Вибір, розробка й обґрунтування елементів технічної забезпечуючої системи	02.06.2025-03.06.2025	Виконано
7	Розробка програмного забезпечення	03.06.2025-06.06.2025	Виконано
8	Тестування та оцінка надійності функціонування програмних і технічних рішень	06.06.2024-06.06.2024	Виконано
9	Синтез і обґрунтування засобів захисту інформації від несанкціонованого доступу	06.06.2024-07.06.2024	Виконано
10	Оформлення пояснювальної записки та графічного матеріалу	07.06.2024-08.06.2024	Виконано
11	Перевірка на плагіат	09.06.2024	Виконано
12	Попередній захист	10.06.2024	Виконано
13	Захист кваліфікаційної роботи в екзаменаційній комісії	12.06.2024	Виконано

Дата видачі завдання 19 травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ ас. Ірина МАЛЬКОВА
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 117 с., 31 рис., 2 табл., 13 джерел, 2 додатки.

АВТОМАТИЗАЦІЯ, БАЗА ДАНИХ, ШТУЧНИЙ ІНТЕЛЕКТ, ІНФОРМАЦІЙНА СИСТЕМА, ІТ-СЕРВІС, ОЦІНКА, ВЕБ-ДОДАТОК, ПРОЕКТ.

Об’єктом дослідження кваліфікаційної роботи є процес відбору пропозицій на розробку ІТ-проектів.

Метою роботи є розробка сервісу аналізу і оцінки пропозицій на розробку ІТ-проектів, який забезпечить високий рівень якості відбору пропозицій, автоматизацію підходів компанії та полегшення роботи її спеціалістів.

Методи дослідження – системний аналіз, засоби моделювання схем даних та веб-орієтовані технології.

У рамках виконання кваліфікаційної роботи проведено аналіз структурних і функціональних особливостей предметної області, огляд і аналіз сучасного стану ІТ-сервісу для відбору пропозицій на розробку ІТ-проектів. В результаті дослідження розроблено концептуальну модель системи, враховуючи організаційні особливості підприємства та його бізнес-процеси.

На основі проведеного аналізу сформульовано перелік ключових вимог до функціоналу системи. Паралельно було спроектовано архітектуру бази даних, що відповідає поставленим завданням.

Фінальним етапом роботи стала практична реалізація веб-застосунку,

основним призначенням якого є автоматизація процесів приймання та аналіз вхідних заявок. Розроблений застосунок повністю відповідає вимогам і забезпечує ефективне вирішення поставлених завдань.

ABSTRACT

The explanatory note to the qualification paper contains: 117 pages, 31 figures, 2 tables, 13 sources, 2 appendix.

AUTOMATION, DATABASE, ARTIFICIAL INTELLIGENCE, INFORMATION SYSTEM, IT SERVICE, EVALUATION, WEB APPLICATION, PROJECT.

The object of the qualification work is the process of selecting proposals for the development of IT projects.

The goal is to develop a service for analyzing and evaluating proposals for the development of IT projects, which will ensure a high level of quality in the selection of proposals, automate the company's approaches and facilitate the work of its specialists.

Research methods – system analysis, data schema modeling tools and web-oriented technologies.

As part of the qualification work, an analysis of the structural and functional features of the subject area was conducted, a review and analysis of the current state of the IT service for the selection of proposals for the development of IT projects was conducted. As a result of the study, a conceptual model of the system was developed, taking into account the organizational features of the enterprise and its business processes.

Based on the analysis, a list of key requirements for the system functionality was formulated. In parallel, a database architecture was designed that meets the tasks set.The final stage of the work was the practical implementation of a web application, the main purpose of which is to automate the processes of acceptance and analysis of incoming applications. The developed application fully meets the requirements and provides an effective solution to the tasks set.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

БД – база даних;

ШІ – штучний інтелект;

ПЗ – програмне забезпечення;

СУБД – система управління базами даних;

API – Application Programming Interface;

QA – Quality Assurance.

ВСТУП

У сучасних умовах стрімкого розвитку цифрових технологій інформаційні системи відіграють ключову роль у забезпеченні стабільного функціонування та стратегічного розвитку компаній, зокрема в сфері інформаційних технологій. Високий рівень конкуренції, потреба в інноваційних рішеннях та постійне оновлення технологій формують нові вимоги до організації внутрішніх процесів в IT-компаніях. Одним із критичних напрямів є процес відбору найбільш перспективних і доцільних пропозицій щодо розробки нових IT-проектів. Такий відбір часто здійснюється вручну або з використанням несистематизованих інструментів, що знижує ефективність прийняття рішень, ускладнює обробку великих обсягів даних і приводить до втрати важливих конкурентних можливостей.

У зв’язку з цим виникає необхідність розробки спеціалізованого сервісу, який автоматизує процеси збору, аналізу та оцінки проектних пропозицій. Даний сервіс має не лише зменшити вплив людського чинника, а й забезпечити прозорість, об'єктивність і надійність у прийнятті управлінських рішень щодо запуску нових IT-проектів. IT-сервіс дозволить формалізувати критерії відбору, зберігати дані про проекти в структурованому вигляді.

Розробка подібного рішення вимагала ретельного аналізу існуючих бізнес-процесів у сфері управління IT-проектами. Було вивчено типові проблеми, з якими стикаються менеджери при оцінці пропозицій, а також визначено ключові вимоги до функціоналу майбутнього сервісу. Важливим аспектом стало забезпечення гнучкості рішення, що дозволяє адаптувати критерії оцінювання під специфічні потреби конкретної організації. Архітектура сервісу була спроектована з урахуванням необхідності інтеграції з існуючими корпоративними інформаційними системами та

забезпечення зручності роботи для всіх категорій користувачів.

Технологічна реалізація передбачає використання сучасних підходів до обробки даних та застосування інструментів штучного інтелекту для підвищення точності аналітичних прогнозів. Сервіс забезпечує автоматизоване ранжування проектів за рівнем пріоритетності, візуалізацію ключових показників у зручному для сприйняття вигляді, а також можливість детального аналізу кожної пропозиції. Особлива увага приділена механізмам захисту даних та розмежуванню прав доступу, що дозволяє забезпечити конфіденційність інформації на всіх етапах роботи з проектами.

Впровадження такого сервісу дозволить ІТ-компаніям істотно підвищити ефективність управління проектним портфелем, зменшити витрати часу на адміністративні процедури та приймати більш обґрунтовані рішення щодо розподілу ресурсів. Крім того, автоматизація процесів відбору створює передумови для більш глибокого аналізу ринкових тенденцій та виявлення перспективних напрямів розвитку. У довгостроковій перспективі це сприятиме формуванню більш збалансованого портфеля проектів, оптимального з точки зору дохідності, інноваційності та відповідності стратегічним цілям організації.

Розробка виконана з урахуванням сучасних вимог до якості програмного забезпечення та передових методологій управління проектами. Запропоноване рішення може бути адаптоване під специфічні потреби різних ІТ-компаній, що робить його універсальним інструментом для вдосконалення процесів відбору проектних ініціатив. Враховуючи постійне зростання кількості пропозицій щодо розробки нових ІТ-рішень, подібні системи стають невід'ємною складовою ефективного управління в технологічному секторі.

1 ЗМІСТОВНИЙ ОПИС ТА АНАЛІЗ СТРУКТУРНИХ І ФУНКЦІОНАЛЬНИХ ОСОБЛИВОСТЕЙ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ОСНОВНИХ ЗАБЕЗПЕЧУЮЧИХ СИСТЕМ

1.1 Аналіз та характеристика об’єкта дослідження

У сучасному цифровому середовищі, де конкуренція за клієнта загострюється, ІТ-компанії зобов’язані реагувати на запити максимально оперативно, забезпечуючи при цьому високу якість аналізу потреб замовника, чітке формулювання завдань та ефективний розподіл ресурсів. Процес реєстрації та аналізу проектних заявок є першим і надзвичайно важливим етапом у життєвому циклі будь-якого ІТ-проекту. Від його ефективності значною мірою залежить загальний успіх реалізації, задоволеність замовника, рівень залученості команди та відповідність результатів очікуванням.

Заявки можуть надходити через різноманітні канали комунікації: електронну пошту, спеціалізовані веб-форми на корпоративному сайті, інтегровані CRM-системи або в результаті ініціатив внутрішніх підрозділів компанії. Зі зростанням кількості запитів зростає також і навантаження на персонал, відповідальний за їх обробку: менеджерів з продажу, бізнес-аналітиків, технічних консультантів тощо. Зазвичай процес включає ознайомлення з описом проекту, з’ясування деталей із замовником, формування попередньої технічної документації, а також оцінку часових та фінансових витрат. Такий підхід ефективний при незначному обсязі вхідних запитів, проте стає нерентабельним і повільним у масштабах великої компанії.

У цьому контексті стає доцільним використання спеціалізованих сервісів для автоматизованого відбору ІТ-пропозицій. Такі системи дозволяють значно оптимізувати процес попереднього аналізу, прискорити прийняття рішень щодо релевантності заявки та розподілу її між

відповідальними фахівцями. Сервіси цього типу, як правило, інтегруються з іншими інформаційними системами організації (CRM, ERP, системами документообігу тощо) й функціонують як єдиний аналітичний інструмент для попередньої фільтрації, класифікації та оцінки запитів.

Об’єктом дослідження в рамках даної кваліфікаційної роботи є процес аналізу та відбору пропозицій на розробку IT-проектів у межах інформаційної системи, що функціонує в середовищі IT-компанії (таблиця 1.1). Даний процес є ключовим компонентом у стратегічному та операційному управлінні діяльністю компанії, яка займається розробкою, впровадженням та супроводом програмних рішень [6].

Призначення сервісу відбору IT-пропозицій полягає у створенні універсальної платформи для централізованого збору та обробки запитів, що забезпечує ефективну аналітику, швидку візуалізацію важливих параметрів та формування релевантних звітів. Завдяки автоматизованому підходу суттєво знижуються ризики людської помилки, покращується точність оцінки проектного потенціалу, а також забезпечується прозорість процесу ухвалення рішень.

Таблиця 1.1 – Загальна характеристика кваліфікаційної роботи

Показник	Опис
Об’єкт автоматизації	Відділ управління проектами
Назва задачі	Розробка сервісу відбору пропозицій на розробку IT-проектів інформаційної системи IT-компанії
Цілі розробки задачі	Створення інтегрованого та ефективного сервісу аналізу і валідації проектів, який забезпечить зручність, швидкість прийняття рішень працівників та підвищить продуктивність роботи компанії.
Призначення задачі	Автоматизація процесу відбору пропозицій

Показник	Опис
Функції задачі	– прийом та обробка заявок; – оцінювання технічної реалізованості проекту; – формування вихідного документу.
Користувачі сервісу	Лідогенератор

Процес відбору заявок на розробку IT-проектів є складною багаторівневою задачею, що вимагає зваженого аналізу великої кількості критеріїв: технічної складності, вартості реалізації, наявності внутрішніх ресурсів, строків, бізнес-пріоритетів та відповідності загальній стратегії розвитку компанії [8]. У сучасних умовах конкуренції та високої динаміки ринку, компанії дедалі частіше звертаються до автоматизованих рішень, що дозволяють підвищити прозорість і об'єктивність процесу, зменшити вплив людського фактора, а також оперативно приймати рішення.

Таким чином, застосування сервісу для відбору IT-пропозицій є стратегічним кроком для IT-компаній, які прагнуть забезпечити масштабованість, оперативність та якість у процесі взаємодії з клієнтами. Такий підхід дозволяє не лише покращити внутрішню ефективність, а й забезпечити вищу конкурентоспроможність на ринку.

1.2 Загальна характеристика IT-компанії «Pecode»

IT-компанія «Pecode» – сучасна українська IT-компанія, яка спеціалізується на розробці веб- та мобільних додатків, а також наданні послуг UI/UX дизайну, QA тестування та підтримки програмних продуктів. Компанія орієнтується на інноваційні рішення для бізнесу, забезпечуючи повний цикл розробки програмного забезпечення для клієнтів із різних

галузей.

Основні напрямки діяльності:

- розробка веб-додатків;
- розробка мобільних додатків;
- UI/UX дизайн;
- тестування програмного забезпечення (QA);
- підтримка та супровід проектів;
- консалтинг із цифрової трансформації бізнесу.

Організаційна структура компанії «Pecode» побудована за матричним типом, що дозволяє ефективно координувати проектну діяльність та оптимізувати внутрішні процеси (рисунок 1.1).

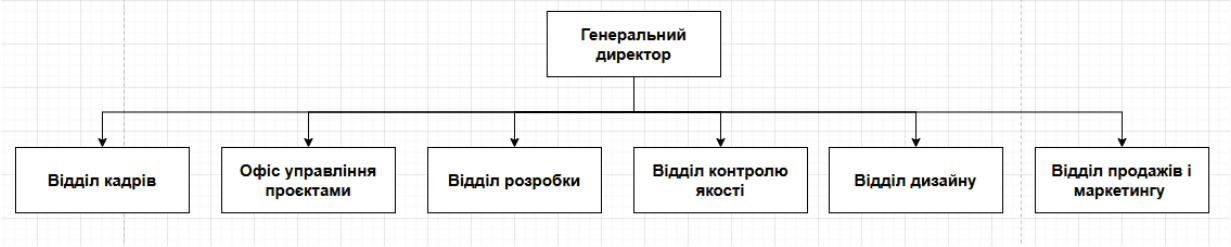

Рисунок 1.1 – Організаційна структура компанії «Pecode»

Об’єктом автоматизаціїї є офіс управління проектами IT-компанії «Pecode».

Офіс управління проектами відповідає за управління проектами, взаємодію з клієнтами, відбір та оцінку пропозицій на розробку IT-проектів. Очолює офіс управління проектами керівник офісу управління проектами (Head of PMO). Ця посадова особа відповідає за стратегічне управління всіма проектами компанії, розробку методології управління проектами, стандартизацію процесів, а також контроль за виконанням завдань у встановлені строки. Керівник офісу управління проектами також взаємодіє з іншими відділами для координації ресурсів і прийняття рішень на високому рівні.

Під керівництвом керівника офісу управління проектами працюють менеджери проектів трьох рівнів.

Старші менеджери проектів (Senior PMs) – мають великий досвід у веденні складних і багаторівневих проектів. Вони беруть участь у стратегічному плануванні, комунікують з ключовими замовниками, формують команди проектів та вирішують ризикові ситуації. Їх обов'язки включають контроль бюджету, строків і досягнення цілей на всіх етапах життєвого циклу проекту.

Менеджери проектів середнього рівня (Middle PMs) – відповідають за управління середніми за складністю проектами. Вони контролюють виконання задач, комунікують із членами команди, формують звіти про стан проекту та слідкують за дотриманням встановлених процедур. Їх завдання також включає оперативне управління змінами та підтримку якісної взаємодії з клієнтом.

Молодші менеджери проектів (Junior PMs) – працюють під наглядом досвідченіших колег, виконують допоміжні функції: підготовка документації, оновлення проектного графіка, організація мітингів, ведення протоколів. Вони навчаються основам управління проектами на практиці, поступово беручи на себе більше відповідальності.

Окрему роль у структурі офісу управління проектами відіграють бізнес-аналітики (Business Analysts). Їх основна функція – збір, аналіз і формалізація вимог від клієнтів або внутрішніх користувачів. Вони формують технічні завдання для розробників, допомагають визначити цілі проекту, проводять аналіз бізнес-процесів і пропонують оптимізаційні рішення. У взаємодії з менеджерами проектів бізнес-аналітики забезпечують відповідність функціоналу розробки очікуванням замовника (рисунок 1.2).

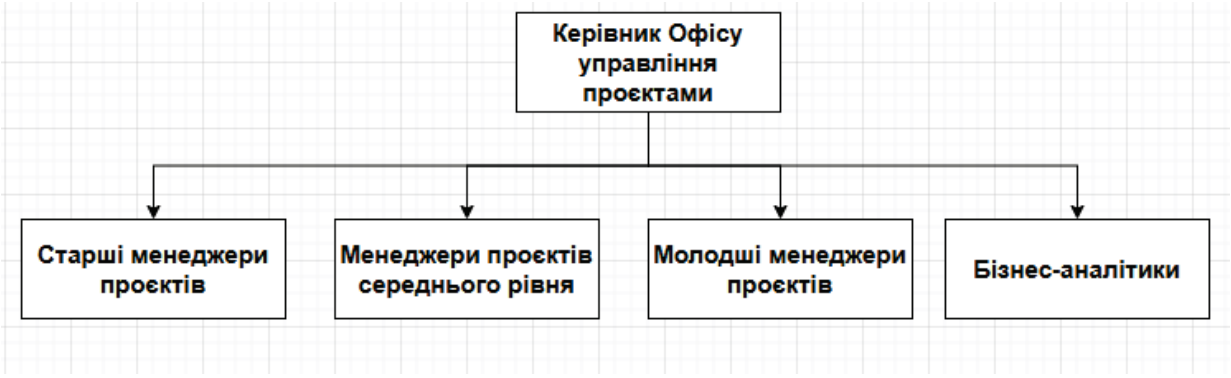

Рисунок 1.2 – Організаційна структура офісу управління проектами

Основні функції офісу управління проектами:

− стандартизація процесів управління проектами: створення та впровадження шаблонів документації, інструкцій, чек-листів;

− контроль ресурсів і строків: моніторинг поточного статусу проектів, виявлення відхилень, оптимізація навантаження команд;

− ризик-менеджмент: аналіз потенційних загроз для реалізації проектів та впровадження заходів із їхнього мінімізації;

− управління якістю: контроль якості виконання задач, відповідність очікуванням клієнта та внутрішнім стандартам компанії;

− підтримка комунікації: забезпечення прозорого обміну інформацією між всіма учасниками проекту.

1.3 Функціональна структура IT-компанії «Pecode»

Функціональна структура визначає основні бізнес-процеси та взаємодію між відділами (рисунок 1.3).

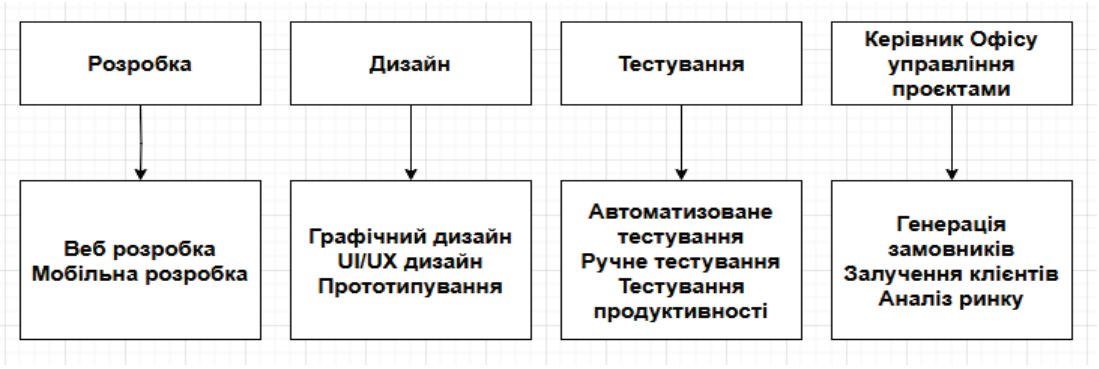

Рисунок 1.3 – Функцінальна структура IT-компанії «Pecode»

Основні бізнес-процеси офісу управління проектами:

- аналіз та відбір пропозицій на розробку проектів;
- планування ресурсів;
- контроль виконання проектів;
- формування звітів для керівництва та клієнтів (рисунок 1.4).

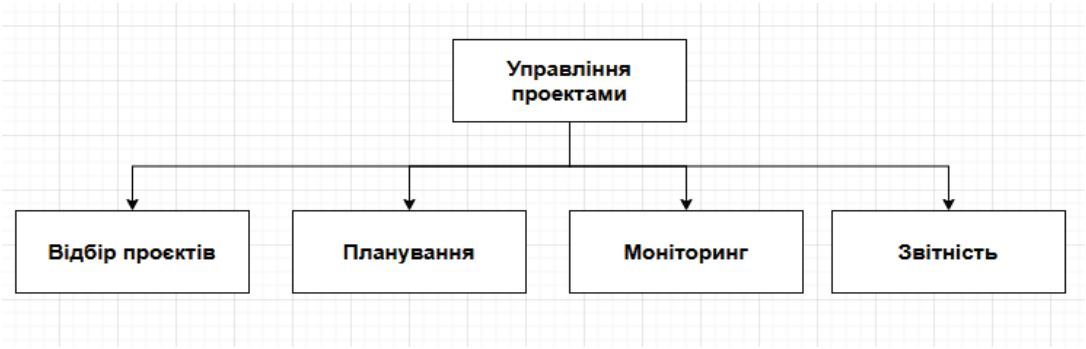

Рисунок 1.4 – Функціональна структура офісу управління проектами

Таким чином, проаналізувавши організаційну та функціональну структуру IT-компанії «Pecode», можна зробити висновок, що вона є ефективно організованим суб'єктом у сфері розробки IT-продуктів, здатним забезпечувати повний цикл створення програмного забезпечення. Впровадження матричної структури управління, чіткий розподіл ролей між менеджерами різного рівня та бізнес-аналітиками, а також наявність офісу управління проектами дозволяють компанії якісно реалізовувати проекти різної складності, забезпечувати взаємодію з клієнтами та відповідати

сучасним вимогам до управління IT-процесами.

1.4 Існуючі проблеми в IT-компанії

У компанії «Pecode» процес обробки проектних пропозицій відбувається за допомогою ручного аналізу або Excel-таблиць , що не інтегруються в загальну інформаційну систему підприємства.

Це зумовлює ряд проблем:

- повторення заявок через відсутність механізмів перевірки на унікальність;
- невизначеність критеріїв оцінювання, що веде до суб’єктивізму;
- нерівномірне навантаження на аналітичні відділи, яке ускладнює управління ресурсами;
- недостатня аналітика щодо причин відхилення заявок;
- відсутність історії комунікації по кожній заявці.

Усе це свідчить про потребу у системному підході до автоматизації процесу, який враховуватиме різноманітні аспекти: від зручності подання до багатофакторного аналізу.

Наявність вищезазначених проблем свідчить про комплексну неефективність традиційного підходу до обробки запитів на розробку IT-продуктів. У зв’язку з цим виникає об’єктивна потреба у розробці спеціалізованого сервісу, що дозволить:

- забезпечити централізований збір заявок з різних джерел у єдиній структурованій системі;
- автоматизувати базові етапи первинного аналізу та класифікації пропозицій;
- стандартизувати критерії оцінки проектів на основі чітко визначених параметрів;

– підвищити прозорість внутрішніх процесів за допомогою логування подій і дій;

– забезпечити якісний та вчасний зворотний зв’язок з клієнтом.

Запровадження такого інтелектуального сервісу сприятиме підвищенню ефективності, зниженню ризиків, економії часу та ресурсів. Крім того, платформа стане основою для накопичення знань, що може бути використано для навчання моделей штучного інтелекту в майбутньому. Це дозволить компанії створити конкурентну перевагу шляхом цифрової трансформації одного з ключових етапів взаємодії з клієнтами – попереднього аналізу їхніх потреб і запитів.

2 ОГЛЯД І АНАЛІЗ СУЧАСНОГО СТАНУ ІТ-СЕРВІСУ ДЛЯ ВІДБОРУ ПРОПОЗИЦІЙ НА РОЗРОБКУ ІТ-ПРОЕКТІВ

2.1 Огляд і аналіз сучасного стану розглянутої проблеми, а також існуючих методів і засобів вирішення задач

Проблема ефективного управління та відбору проектних пропозицій на етапі їх подання є однією з найактуальніших у сфері сучасного ІТ-бізнесу. В умовах постійного зростання попиту на розробку цифрових рішень, компанії отримують десятки, а іноді й сотні заявок щомісячно. При цьому якість, повнота та бізнес-потенціал цих заявок суттєво відрізняються, що створює виклики для оперативної та об'єктивної аналітики.

Багато компаній намагаються впровадити рішення, які б забезпечували баланс між автоматизацією процесу та збереженням контролю за ключовими етапами аналізу. Проте часто на практиці застосовуються застарілі або малоефективні підходи, які не відповідають вимогам часу [6].

2.2 Аналіз традиційних і сучасних підходів

Основними підходами до обробки заявок на розробку ІТ-проєктів є:

- ручна експертна оцінка– такий підхід домінував в ІТ-компаніях до середини 2010-х років. Він передбачає особистий перегляд і аналіз кожної заявки відповідними фахівцями: проєктними менеджерами, технічними лідерами, бізнес-аналітиками. Хоча цей метод дозволяє враховувати специфіку проєкту, він є надзвичайно ресурсоємним і суб’єктивним. При великій кількості заявок об’єктивність оцінки часто знижується через втому або брак часу, а прийняті рішення можуть бути непослідовними;
- CRM-рішення з базовою автоматизацією – інтеграція процесу

подання заявок у CRM-системи (наприклад, Bitrix24, Zoho, HubSpot) дозволяє структурувати дані, проте лише частково вирішує проблему оцінки. Такі системи часто використовуються лише як сховище інформації, тоді як її інтерпретація залишається поза межами автоматизації. Вони не мають достатньої гнучкості для складної багатофакторної аналітики;

– ERP-рішення корпоративного рівня (SAP, Oracle) – інтеграція процесів відбору у великі ERP-системи можлива і практикується у транснаціональних корпораціях. Такі рішення включають побудову складних бізнес-процесів, автоматизоване делегування завдань, логіку погодження тощо. Проте їхній основний недолік – висока вартість впровадження, складність адаптації до змін, а також надмірність функціоналу для середніх і невеликих IT-компаній;

– власні веб-платформи – останнім часом зростає популярність індивідуально розроблених веб-сервісів для внутрішнього користування. Такі рішення дозволяють реалізовувати специфічні алгоритми фільтрації, скорингу, категоризації заявок. Це особливо актуально для компаній, які прагнуть швидко адаптувати систему до змін у структурі бізнесу або запуску нових напрямів.

Сучасні технології, що використовуються в автоматизованих системах:

– BPMN (Business Process Model and Notation) – дозволяє візуалізувати процес подання, погодження та відбору заявок, що особливо корисно для оптимізації взаємодії між підрозділами;

– штучний інтелект (AI) – використовується для попереднього аналізу вхідної інформації, визначення потенційної складності проекту, виявлення прихованих ризиків;

– машинне навчання (ML) – дає змогу будувати моделі, що на основі історичних даних прогнозують імовірність успішного завершення проекту або його фінансової ефективності;

– системи електронного документообігу – дозволяють формалізувати

погодження, перевірку документів, зберігання цифрових підписів та інших юридично значимих процедур;

– BI-платформи (Business Intelligence) – Power BI, Tableau, Google Data Studio – використовуються для побудови звітів, графіків, моніторингу динаміки заявок та ефективності їх розгляду.

Попри існуюче різноманіття технологічних рішень, більшість із них не повністю відповідають потребам середніх IT-компаній. Серед типових обмежень:

– надмірна складність інтерфейсів;

– висока вартість підтримки;

– низький рівень кастомізації без додаткового програмування;

– труднощі в інтеграції з існуючими модулями компанії;

– орієнтація на великі компанії з усталеними структурами, що ускладнює адаптацію у динамічних проектних командах.

Таким чином, існуючі підходи лише частково вирішують проблему ефективного управління заявками на IT-проекти. Найбільш перспективним шляхом є створення власного сервісу, адаптованого під конкретні потреби організації. Це дозволяє уникнути залежності від зовнішніх платформ, досягти гнучкості, швидко вносити зміни у процеси та зосередитися на підвищенні ефективності внутрішньої взаємодії. Власний сервіс із модульною архітектурою забезпечує і високу адаптивність, і технічну еволюційність системи [7].

У діяльності IT-компанії «Pecode» було виявлено низку організаційних і технологічних проблем, пов’язаних з процесом відбору пропозицій на розробку IT-проектів. Попри використання сучасних інструментів управління (таких як Jira, Trello, Slack, Google Workspace), значна частина процесу первинної оцінки клієнтських запитів виконується вручну, що безпосередньо впливає на ефективність компанії.

Наразі у IT-компанії «Pecode» використовується комбінація наступних інструментів:

− таблиці Excel або Google Sheets для занесення вхідних запитів від клієнтів;

− електронна пошта та месенджери (Slack, Telegram) для обміну інформацією між Sales-менеджерами, аналітиками та Project-менеджерами;

− Zoom/Meet – для обговорення кожного потенційного проекту вручну;

− Jira/Trello – для управління вже затвердженими проектами, але не для первинного відбору.

− Такий підхід є ефективним лише за умов невеликого навантаження, однак у міру зростання кількості запитів виявляються істотні недоліки існуючої системи:

− – велике навантаження на менеджерів і аналітиків, які змушені витрачати час на фільтрацію запитів вручну;

− високий ризик людських помилок, що призводить до втрати важливих заявок або дублювання даних;

− відсутність єдиного централізованого сховища інформації, що ускладнює зведення статистики або аналіз історії рішень;

− повільна реакція на нові запити, через необхідність погодження в кілька етапів вручну;

− неможливість відстеження прогресу в реальному часі, а також складність у формуванні звітів для керівництва.

Усе це створює додаткові організаційні ризики: компанія може втрачати потенційно вигідні проекти, не встигнувши оперативно їх оцінити; команда може перевантажуватися рутинною роботою, що не приносить цінності, але займає час і ресурси.

Саме в цьому контексті виникає необхідність у впровадженні спеціалізованого IT-сервісу, який автоматизує процес відбору пропозицій.

Цей сервіс має забезпечити:

− збір і структурування вхідних заявок в одній системі;

− автоматичне сортування за критеріями (технології, бюджет, термін,

складність);

- швидке формування звітів та історії взаємодії;
- можливість оцінки ефективності відбору проектів на основі даних.

Таким чином, розробка власного сервісу є не лише рішенням локальної проблеми, а й стратегічним кроком до оптимізації бізнес-процесів компанії «Pecode» , що дозволяє:

- підвищити продуктивність команди;
- скоротити час на прийняття рішень;
- зменшити ризики помилок;
- покращити якість обслуговування клієнтів.

2.3 Аналіз існуючих аналогів

У процесі розробки власного сервісу для автоматизованого відбору пропозицій на розробку IT-проектів важливим етапом є аналіз наявних рішень, які вже існують на ринку. Такий аналіз дозволяє не тільки оцінити функціональні можливості конкурентів, але й визначити їхні сильні та слабкі сторони, що у свою чергу створює підґрунтя для розробки унікального й конкурентоспроможного продукту. У контексті дослідження розглядаються як безпосередні конкуренти-платформи, що спеціалізуються на керуванні заявками чи тендерами на IT-послуги, так і суміжні сервіси, які мають часткову функціональність, релевантну тематиці розробки.

2.3.1 Upwork та інші фриланс-платформи

Одним із найбільш відомих аналогів, хоча й з іншим фокусом, є

платформи типу Upwork, Freelancer, Toptal. Ці сервіси надають можливість замовникам створювати проекти, а виконавцям – подавати заявки. У контексті відбору пропозицій, тут також реалізовано певний механізм фільтрації, який ґрунтується на рейтингах, досвіді, відгуках, навичках та іншій інформації про виконавців. Однак ці платформи працюють у відкритому ринковому середовищі й не орієнтовані на внутрішні процеси компанії. Їхні алгоритми не мають можливості глибокого налаштування відповідно до індивідуальних критеріїв IT-компанії (рисунок 2.1).

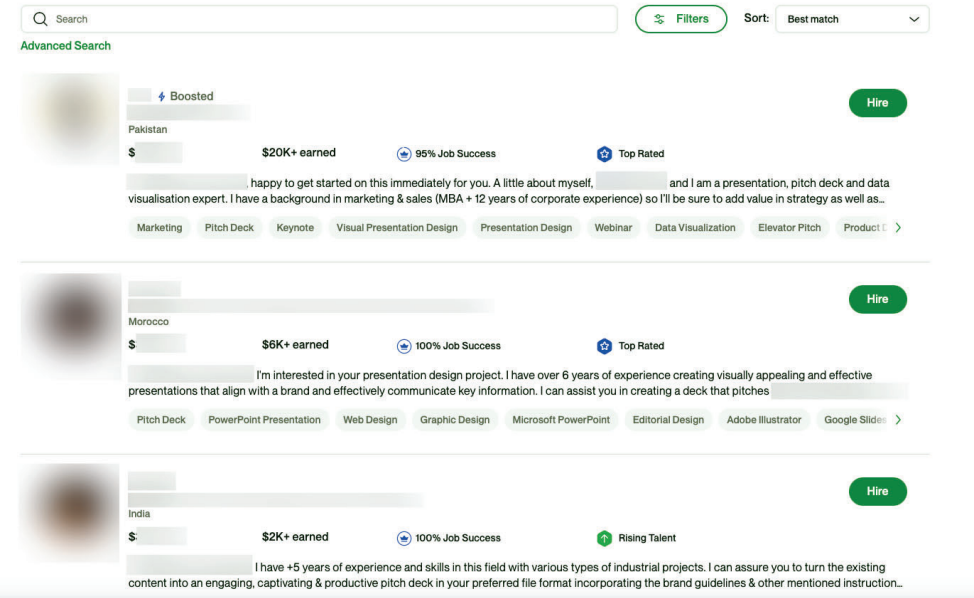

Рисунок 2.1 – Список заявок Upwork

Також бракує гнучких засобів інтеграції з внутрішніми ERP/CRM-системами, аналітики для стратегічного планування та контролю за ефективністю прийнятих рішень у контексті корпоративних цілей. Таким чином, хоча Upwork та подібні сервіси можна вважати аналогами, вони не є прямими конкурентами нашого проекту, оскільки мають іншу бізнес-модель і орієнтовані на зовнішнє, а не внутрішнє використання.

2.3.2 Jira + Jira Service Management

У багатьох IT-компаніях для обробки заявок використовуються інструменти на базі Jira та Jira Service Management, які дозволяють організувати процес прийому запитів на розробку, зберігати історію змін, призначати відповідальних і контролювати виконання. Проте, хоча ці системи дуже гнучкі, їх налаштування для автоматичного прийняття рішень або скорингового аналізу вимагає значних зусиль і часто потребує кастомної розробки (рисунок 2.2).

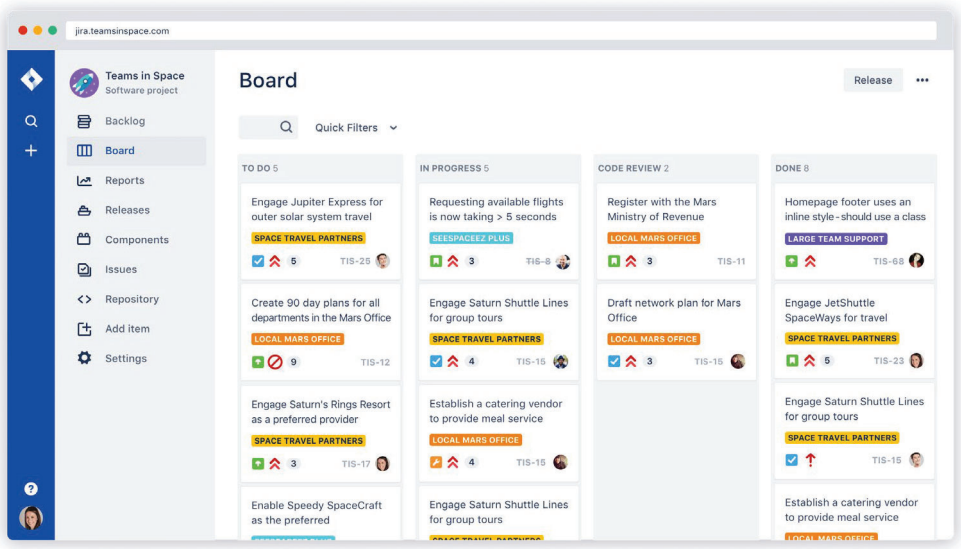

Рисунок 2.2 – Зовнішній вигляд системи контролю Jira

Jira – це, в першу чергу, інструмент для керування завданнями та інцидентами, а не для автоматизованого відбору проектних заявок. У більшості випадків вона працює як система реєстрації та контролю, а не як інтелектуальний фільтр. До того ж відсутній механізм комплексного оцінювання заявок за різними критеріями, такими як рентабельність, наявність ресурсів, пріоритетність, відповідність стратегічним цілям тощо.

2.3.3 Monday.com, Asana, ClickUp

Популярні SaaS-рішення для керування проектами – Monday.com, Asana, ClickUp – також мають модулі для подання заявок та їх обробки. Зокрема, вони дозволяють створювати форми, збирати пропозиції, будувати пайплайни та призначати задачі командам. У деяких із них навіть присутня інтеграція з аналітичними інструментами або можливість візуалізації даних у вигляді діаграм і таблиць (рисунок 2.3).

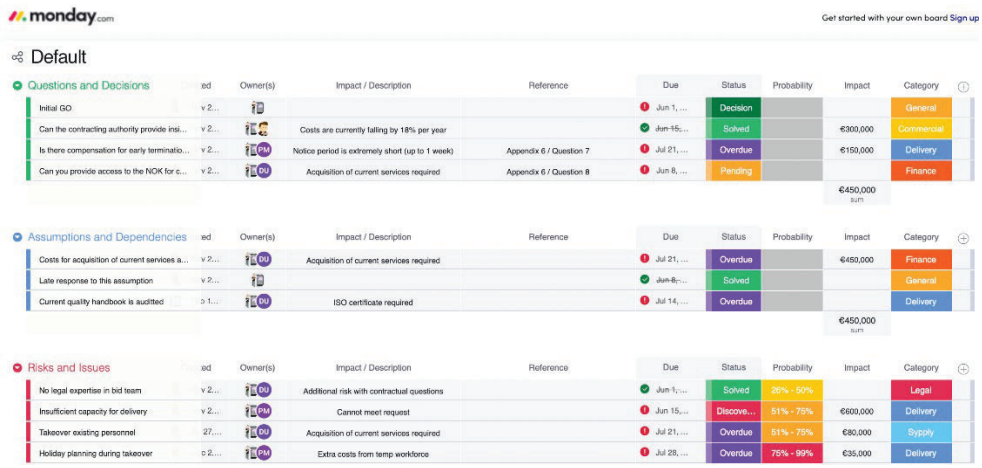

Рисунок 2.3 – Зовнішній вигляд системи контролю Monday.com

Однак, як і у випадку з Jira, ці інструменти більше фокусуються на управлінні задачами, а не на прийнятті рішень щодо відбору ініціатив. Відсутня спеціалізована логіка для відсіву малоперспективних заявок, а також не підтримується адаптивна система оцінювання заявок за множиною параметрів. Іншими словами, вони не є повноцінними системами попереднього аналізу та пріоритезації, а радше інструментами для реалізації вже схвалених проектів.

2.3.4 Airtable + кастомні системи

Деякі IT-компанії використовують Airtable або розробляють власні внутрішні рішення на базі Google Forms, Excel Online, Notion або кастомних веб-застосунків. У таких системах подання заявок реалізується у вигляді форм, а відбір здійснюється вручну або через частково автоматизовані правила (рисунок 2.4).

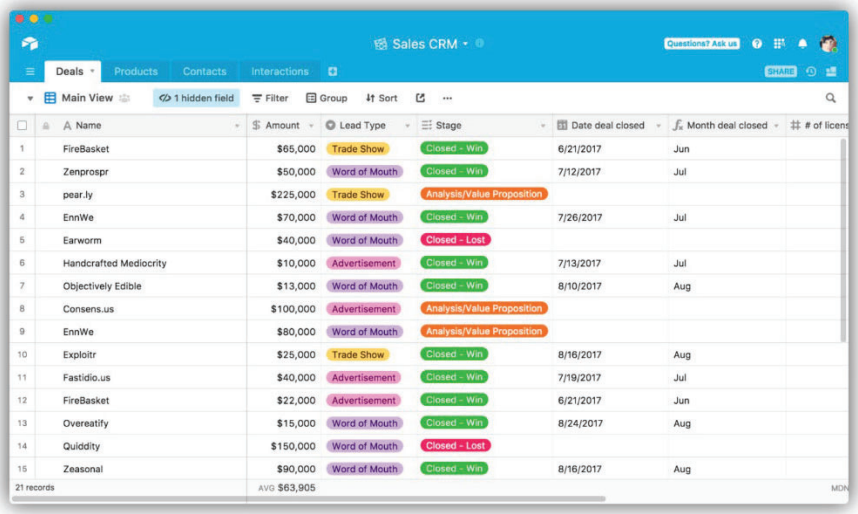

Рисунок 2.4 – Зовнішній вигляд системи контролю AirTable

Перевагою таких рішень є швидкість впровадження та мінімальні витрати на початковому етапі. Проте масштабування таких систем, інтеграція з зовнішніми сервісами, гарантія безпеки та централізоване зберігання історії змін часто залишаються поза увагою. Крім того, кастомні рішення зазвичай не мають вбудованої логіки оцінювання ефективності проектів чи формування аналітичної звітності, що критично важливо для великих компаній.

Отже, на основі порівняння можна зробити висновок, що наявні рішення або орієнтовані на зовнішній ринок (Upwork, Freelancer), або є універсальними інструментами керування завданнями (Jira, Asana,

Monday.com), але не мають спеціалізованої логіки, адаптованої до потреб внутрішнього попереднього відбору заявок в IT-компаніях.

У зв’язку з цим виникає потреба у створенні окремого сервісу, що поєднує:

- адаптивну систему оцінювання заявок;
- можливість інтеграції з внутрішніми корпоративними системами;
- автоматизоване прийняття рішень на основі множини параметрів;
- інструменти звітності та стратегічної аналітики.

Це дозволить компаніям зосередитися на реалізації найбільш перспективних ініціатив, оптимізувати витрати ресурсів і сформувати прозору та масштабовану модель прийняття управлінських рішень у межах процесу розробки IT-проектів.

3 ФОРМУЛЮВАННЯ ЗАВДАННЯ РОЗРОБКИ

3.1 Опис вимог до об'єкта розробки

Поставленою задачею роботи є проведення аналізу сучасних тенденцій у сфері автоматизації процесів управління ІТ-проектами, а також розробка - ІТ-сервісу, який забезпечить ефективний відбір пропозицій на реалізацію програмних рішень для ІТ-компаній. У контексті динамічного розвитку цифрових технологій і постійного зростання попиту на інноваційні ІТ-послуги, ключовим завданням є забезпечення швидкого, надійного та об'єктивного механізму прийняття рішень щодо того, які проекти доцільно реалізовувати в першу чергу. Враховуючи обмежені ресурси компаній – фінансові, людські та часові – критично важливим є впровадження спеціалізованих інструментів, що дозволяють систематизувати запити, здійснювати їхню якісну оцінку, а також формувати прозору логіку вибору пріоритетних напрямків розробки.

У сучасному інформаційному просторі щоденно формується значна кількість запитів на створення ІТ-рішень, що змушує компанії шукати способи автоматизації процесу їх обробки. Без відповідного механізму обліку, аналізу та ранжування проектних заявок зростає ризик неефективного використання ресурсів, втрати потенційно вигідних контрактів або, навпаки, запуску невиправдано витратних ініціатив. Саме тому на передній план виходить завдання створення універсального інструменту, який не лише оптимізує управлінську діяльність, але й сприятиме досягненню високого рівня обґрунтованості прийнятих рішень.

Метою цієї кваліфікаційної роботи є розробка інтелектуального ІТ-сервісу, що дозволяє автоматизувати процес обробки пропозицій на розробку ІТ-проектів. Основна функціональність сервісу має забезпечити інтерактивну подачу заявок, автоматичну їхню оцінку згідно з визначеними критеріями, формування аналітичних висновків і візуалізацію результатів з

можливістю інтеграції в існуючі корпоративні інформаційні системи. Такий сервіс повинен бути зручним у користуванні, мати інтуїтивно зрозумілий інтерфейс та відповідати сучасним вимогам щодо продуктивності, масштабованості, інформаційної безпеки й гнучкості архітектури.

Для досягнення поставленої мети передбачається реалізація наступних ключових завдань:

− розробка інтерфейсу для подання заявок користувачами. Цей елемент повинен забезпечувати можливість введення детальної інформації про потенційний проект, включаючи опис проблеми, очікувані результати, бюджетні обмеження, часові рамки та стратегічні цілі;

− створення алгоритму автоматизованої оцінки та скорингу проектних заявок. Передбачається впровадження системи критеріїв оцінювання з урахуванням вагових коефіцієнтів, які дозволять надавати об'єктивну і релевантну оцінку кожній пропозиції на основі множини параметрів;

− реалізація модуля управління життєвим циклом заявок. Це передбачає можливість змінювати статуси заявок, вести їх облік, зберігати історію змін і надавати доступ до інформації різним ролям користувачів (наприклад, аналітик, керівник проекту, замовник);

− розробка модуля аналітики та візуалізації даних. Такий компонент дозволить здійснювати поглиблений аналіз тенденцій, виявляти закономірності у поданих пропозиціях, прогнозувати ризики та підтримувати стратегічне планування;

− інтеграція сучасних підходів до захисту даних та масштабування. Планується використання хмарних рішень та контейнеризованих середовищ, що забезпечать гнучкість у розгортанні та адаптації до змін у структурі навантаження.

Не менш важливим аспектом реалізації задачі є правильний вибір технологічного стека. Для створення ефективного, гнучкого й масштабованого рішення пропонується використання React.js для

клієнтської частини, Node.js – для реалізації серверної логіки, PostgreSQL як систем управління базами даних. Додатково, з огляду на необхідність інтелектуальної обробки заявок, доцільно інтегрувати інструменти штучного інтелекту (ШІ), які дозволять реалізувати функції машинного навчання, класифікації, прогнозування й автоматичного фільтрування заявок.

Таким чином, результатом реалізації даного проекту стане не лише створення корисного програмного забезпечення, а й впровадження концептуального підходу до організації процесу прийняття рішень у сфері IT-проектування, що здатен підвищити ефективність бізнес-процесів компанії та зміцнити її конкурентні позиції на ринку.

Також, у сучасному етапі розвитку IT-галузі особливої актуальності набуває використання готових рішень на основі штучного інтелекту, які дозволяють значно зменшити витрати часу та ресурсів на розробку власних алгоритмів з нуля. У рамках реалізації сервісу відбору пропозицій на розробку IT-проектів доцільним є використання вже протестованих та ефективних AI-модулів, які забезпечують високу точність класифікації, ранжування, аналізу та прогнозування результатів.

Зокрема, можливим є впровадження наступних рішень:

– AI-модулі для автоматичного скорингу заявок, що базуються на алгоритмах класифікації (наприклад, XGBoost, CatBoost, нейронні мережі). Такі модулі здатні визначати ймовірність успішності проекту на основі історичних даних та попередніх реалізованих кейсів, що дозволяє формувати більш обґрунтовані управлінські рішення [9];

– інструменти для обробки природної мови (NLP), які забезпечують автоматичний аналіз описів заявок: виділення ключових слів, виявлення емоційного тону, тематичне кластерування. Це дає можливість проводити семантичне порівняння нових запитів з базою успішних або неуспішних проектів, що вже реалізовувалися компанією;

– AI-помічник для автоматичної генерації відповідей та звітів, який

може формувати звіти для менеджерів або клієнтів на основі контексту заявки. Це суттєво зменшує навантаження на аналітиків і сприяє стандартизації відповіді;

− готові аналітичні модулі, наприклад, Google AutoML або Microsoft Azure Cognitive Services, які дозволяють без глибоких знань у сфері машинного навчання інтегрувати передові AI-інструменти до внутрішніх бізнес-процесів компанії. Вони можуть забезпечити зручну візуалізацію даних, побудову дашбордів, динамічну оцінку ефективності проектів у реальному часі [7];

− системи звітів, що використовуються для порівняння нових заявок із вже успішно реалізованими проектами, ідентифікації схожих кейсів та пропозиції релевантних рішень. Такі моделі часто базуються на колаборативній фільтрації або гібридних підходах [10].

Інтеграція готових AI-рішень дає змогу зосередитися не стільки на розробці базових алгоритмів, скільки на адаптації їх до специфіки внутрішніх бізнес-процесів. При цьому суттєво скорочується час розробки MVP (мінімально життєздатного продукту) та знижується ризик технічних помилок у ранніх етапах впровадження. Крім того, більшість сучасних AI-платформ підтримують масштабованість, хмарну інтеграцію та мають широкий спектр API, що спрощує їх включення до архітектури проекту.

Таким чином, впровадження готових рішень на основі штучного інтелекту є не лише технічно доцільним, а й стратегічно важливим кроком у побудові конкурентоспроможного, надійного та інноваційного продукту для IT-компаній. Завдяки цьому сервіс отримає суттєві переваги в плані точності оцінки, швидкості обробки заявок, аналітичного потенціалу та зручності для кінцевих користувачів.

3.2 Функціональні вимоги

У рамках розробки сервісу відбору пропозицій на розробку IT-проектів особливе значення мають функціональні вимоги, які визначають набір основних задач, що повинна виконувати сервіс для ефективного функціонування. Основна мета полягає в реалізації повного циклу обробки пропозицій, від моменту їх створення до прийняття остаточного рішення щодо доцільності реалізації. Функціональність сервісу охоплює як базові механізми взаємодії з користувачем, так і складні алгоритми аналітичної обробки даних.

Першим етапом взаємодії користувача з сервісом є створення та подання нової пропозиції. Інтерфейс дозволяє заповнити інтерактивну форму, де зазначаються основні параметри: назва майбутнього проекту, його докладний опис, обраний технологічний стек, бажаний бюджет і термін виконання. Уся ця інформація становить базу для подальшої автоматизованої обробки.

Одразу після подання заявка проходить попередній технічний аналіз, який включає інтелектуальне зіставлення запропонованого стеку технологій з внутрішніми базами знань компанії. Сервіс порівнює обрані інструменти з наявними компетенціями фахівців, що дозволяє швидко визначити відповідність між вимогами проекту і технічними можливостями команди. Наступним кроком є перевірка фінансової складової: запропонований бюджет оцінюється шляхом порівняння з внутрішніми розцінками, що дозволяє виявити економічну обґрунтованість реалізації пропозиції.

Сервіс також автоматично оцінює доступність необхідних ресурсів. Це включає аналіз наявності фахівців із відповідною кваліфікацією, врахування їхньої поточної зайнятості та формування попередньої оцінки термінів виконання. На основі отриманих даних сервіс формує звіт з оцінки

технічної реалізованості проекту.

У разі необхідності додаткової перевірки сервіс передбачає передачу пропозиції технічному експерту. Він має змогу підтвердити запропоноване рішення або внести корективи на основі глибшого аналізу. Такий підхід забезпечує баланс між автоматизацією процесу і контролем з боку спеціалістів.

Для підтримки актуальності даних реалізовано функціонал редагування внутрішніх таблиць та баз знань. Адміністратор сервісу має доступ до конфігураційних параметрів, таких як списки технологічних стеків, профілі спеціалістів, внутрішні ставки, що дозволяє швидко адаптувати сервіс до змін в організації.

Не менш важливою є наявність потужного модуля автентифікації та авторизації, який відповідає за безпеку й контроль доступу. Сервіс підтримує реєстрацію користувачів із підтвердженням електронної пошти, вхід за допомогою email/паролю, а також інтеграцію з системами єдиного входу (SSO). Ролі користувачів чітко розмежовані – замовники, аналітики, адміністратори, ревізори – що дозволяє організувати ієрархічну модель доступу.

Формування та обробка заявок реалізовані через адаптивні форми, які динамічно реагують на введені дані, підвантажуючи необхідні поля в залежності від контексту. Користувачі можуть редагувати подані заявки в межах наданих прав, а також видаляти або архівувати їх із підтвердженням дій, що забезпечує безпеку та контроль за даними.

Однією з ключових переваг сервісу є впровадження багатокритеріальної моделі оцінювання заявок. Кожна пропозиція аналізується за низкою параметрів: технічна складність, комерційна доцільність, часові обмеження, ресурсозатратність. Після введення всіх необхідних значень сервіс автоматично генерує звіт з оцінки технічної реалізованості проекту.

Таким чином, сукупність описаних функцій дозволяє створити

ефективну, масштабовану та безпечний сервіс відбору пропозицій на розробку IT-проектів, яка враховує всі аспекти прийняття рішень – від технічних до організаційно-управлінських.

3.3 Нефункціональні вимоги

Нефункціональні вимоги визначають якісні характеристики інформаційної системи, які не пов’язані безпосередньо з реалізацією функціональних сценаріїв, але критично важливі для стабільної, ефективної та безпечної роботи користувачів. У розроблюваному сервісі відбору пропозицій на розробку IT-проектів вони охоплюють такі аспекти, як продуктивність, масштабованість, безпека, кросплатформеність, надійність, інтернаціоналізація, UX/UI-якість та архітектурна гнучкість.

Передусім сервіс має забезпечувати високу швидкодію, тобто обробку запитів, включно з аналізом пропозицій, у межах 1–3 секунд при стандартному навантаженні. Це дозволяє користувачам оперативно отримувати результати аналізу заявок і приймати управлінські рішення без затримок.

Масштабованість – ще один ключовий критерій. Архітектура сервісу спроектована таким чином, щоб у разі збільшення обсягу даних або навантаження, система могла бути розширена без суттєвих змін у логіці. Передбачено як горизонтальне масштабування серверів, так і можливість інтеграції додаткових модулів, включно з API зовнішніх сервісів – наприклад, біржі фрилансу Upwork. Таке розширення дозволить отримувати нові проектні пропозиції автоматично, без ручного внесення інформації.

Забезпечується крос-браузерна підтримка (Chrome, Firefox, Safari, Edge), а також адаптивність інтерфейсу під мобільні пристрої, що робить сервіс зручним для використання на різних платформах. Інтерфейс

користувача підтримує інтернаціоналізацію, зокрема можливість перемикання мов (українська, англійська).

З архітектурної точки зору, сервіс є багаторівневим веб-застосунком, який інтегрується у ширшу інформаційну екосистему IT-компанії. Його структуру можна умовно поділити на кілька функціональних модулів:

- модуль подання заявок – забезпечує користувацький інтерфейс для створення структурованих заявок на розробку проектів, інтегрований із системою автентифікації;
- модуль первинної валідації – перевіряє заповненість форми, відповідність встановленим вимогам, відсіює дублікатні заявки;
- модуль експертної оцінки – відповідає за надання заявкам рейтингової оцінки або ініціює запити до аналітиків та технічних експертів для детального аналізу;
- модуль аналітики – дозволяє генерувати звіти, виявляти закономірності, будувати графіки й дашборди для менеджменту;

Для реалізації цього функціоналу застосовується сучасний стек технологій: на клієнтській частині – React або Next.js, на серверній частині – Node.js або Django, у якості бази даних – PostgreSQL. Особливий акцент робиться на API-орієнтованій архітектурі, яка забезпечує зручну інтеграцію з мобільними застосунками, зовнішніми сервісами та аналітичними платформами.

Таким чином, нефункціональні вимоги до сервісу не тільки створюють основу для ефективної реалізації функціональності, а й забезпечують масштабованість, стабільність, безпеку та гнучкість, що критично важливо для підтримки життєвого циклу цифрового продукту в умовах швидкозмінного IT-середовища.

3.4 Обґрунтування мети і критеріїв ефективності об’єкта розробки

Основною метою розробки даного сервісу є створення гнучкого, автоматизованого інструменту, який дозволяє значно підвищити якість процесу відбору проектів на ранньому етапі – при надходженні заявки. Розробка повинна сприяти поліпшенню внутрішньої комунікації в IT-компанії, підвищенню прозорості процесів ухвалення рішень і зменшенню людського фактору в процесі прийняття критично важливих бізнес-рішень [8].

Очікувані результати:

- скорочення часу на аналіз заявки завдяки автоматичному оцінюванню та структурованому зберіганню інформації;
- зменшення рівня суб'єктивізму в процесі ухвалення рішень за рахунок багатофакторного аналізу;
- оптимізація ресурсів компанії – як часових, так і кадрових;
- зменшення кількості відхилених проектів на пізніх етапах реалізації;
- покращення показників реалізації успішних IT-проектів – підвищення їхньої прибутковості та якості.

Ключові критерії ефективності:

- кількість заявок, опрацьованих сервісом за обраний період (продуктивність);
- зменшення кількості ревізій заявок на етапах аналізу та погодження;
- кількість виявлених ризиків, які були вчасно оброблені завдяки сервісу;
- зниження навантаження на команду аналітиків – зменшення рутинної роботи.

4 ОПИС АРХІТЕКТУРИ ОБ'ЄКТА РОЗРОБКИ НА РІВНІ ФУНКЦІЙ

4.1 Опис вимог до програмного сервісу

У процесі створення програмного забезпечення надзвичайно важливо на початкових етапах чітко та структуровано визначити вимоги до майбутнього сервісу. Це дозволяє уникнути непорозумінь, забезпечує цілісність реалізації та полегшує тестування й підтримку. У рамках розробки IT-сервісу «Відбір пропозицій на розробку IT-проектів» було сформовано як функціональні, так і нефункціональні вимоги, які визначили основні принципи побудови архітектури, технологічний стек, модель взаємодії користувачів з інтерфейсом, а також критерії ефективності роботи сервісу.

Основна мета проекту полягає в автоматизації процесу первинного аналізу проектних пропозицій, що надходять з бірж фрилансу (зокрема таких як Upwork, Freelancer, Toptal тощо), для подальшого прийняття рішення щодо їх релевантності потребам та можливостям IT-компанії. Часто такі пропозиції вимагають ручного перегляду та аналізу, що віднімає значний обсяг робочого часу в фахівців, зокрема в лідогенераторів, які відповідають за пошук перспективних проектів.

Лідогенератор – це ключова особа у цьому процесі, адже саме він здійснює первинне ознайомлення з проектом, перевіряє його відповідність внутрішнім технічним і організаційним стандартам компанії, а також визначає, чи є сенс інвестувати ресурси в подальшу участь у тендері. Щодня лідогенератори можуть переглядати від кількох десятків до сотні пропозицій, і навіть незначна оптимізація цього процесу значно впливає на загальну продуктивність і якість відбору.

У зв'язку з цим перед сервісом було поставлено завдання: максимально автоматизувати рутинні етапи аналізу пропозицій, надавши інтелектуальну підтримку у формі веб-додатку, який виконує попередню

перевірку запитів за заданими критеріями та генерує аналітичні висновки. Основні переваги такого підходу:

− скорочення часу на первинну обробку вхідних запитів завдяки автоматизованому аналізу;

− підвищення точності при виборі пропозицій, які відповідають внутрішнім вимогам компанії (у тому числі за бюджетом, строками, стеком технологій тощо);

− формалізація та централізація логіки прийняття рішень, що знижує ризики людського фактору;

− можливість накопичення статистичних даних, що дає змогу будувати прогностичні моделі та покращувати бізнес-аналітику.

4.1.1 Архітектура сервісу

З огляду на сучасні вимоги до веб-розробки, сервіс було спроектовано як класичний клієнт-серверний додаток, де клієнтську і серверну частини розділено, але тісно пов’язано через API. Такий підхід гарантує масштабованість, можливість повторного використання модулів та простоту в обслуговуванні.

Клієнтська частина реалізована з використанням бібліотеки React.js, яка забезпечує високу продуктивність, інтерфейсну гнучкість та швидку реакцію на дії користувача. Завдяки компонентній структурі, інтерфейс можна швидко модифікувати або розширювати без ризику для цілісності всього проекту.

Серверна частина побудована на платформі Node.js, яка дозволяє реалізовувати асинхронну обробку запитів і працювати з великою кількістю паралельних запитів у режимі реального часу. Сервер відповідає за обробку бізнес-логіки, взаємодію з базами даних, запуск аналітичних алгоритмів і

повернення відповідей у структурованому форматі (зазвичай JSON).

База даних реалізована на основі PostgreSQL – потужної об'єктно-реляційної системи управління базами даних, яка забезпечує збереження інформації про розробників, проекти, бюджети, історію запитів, попередні рішення та інші критично важливі дані.

4.1.2 Основні компоненти сервісу

Сервіс поділено на низку функціональних блоків, кожен з яких відповідає за певний етап обробки запиту:

- інтерфейс введення даних – забезпечує користувача зручною формою для внесення параметрів проекту (назва, короткий опис, бюджет, терміни, технології тощо). Це перша точка взаємодії, де формується вхідна заявка;
- модуль аналізу релевантності – автоматично співставляє параметри вхідної заявки з внутрішніми критеріями компанії. Зокрема, аналізується відповідність технологічного стеку, доступність потрібних спеціалістів, середні ставки та інші фактори;
- модуль оцінки та аналізу заявки – проводить попередню оцінку строків виконання, виходячи з обсягу задач, складності реалізації та поточної завантаженості команди;
- модуль формування звіту – результати подаються в зручному візуальному форматі для подальших дій користувача.

4.1.3 Алгоритм функціонування сервісу

Нижче подано послідовний опис роботи сервісу з моменту введення пропозиції до отримання аналітичного висновку:

- користувач (лідогенератор) через веб-інтерфейс вносить нову пропозицію до сервісу;
- сервіс автоматично ідентифікує технологічний напрям (наприклад, Frontend, Backend, iOS, Android тощо) на основі ключових слів і тегів;
- відбувається перевірка відповідності: чи є в компанії доступні спеціалісти для виконання запропонованого проекту;
- аналізується запропонований бюджет у порівнянні з внутрішніми розцінками на відповідні послуги;
- перевіряється доступність команди на заявлені строки реалізації;
- формується попередній прогноз щодо строків виконання завдання;
- на основі всіх факторів сервіс генерує звіт з оцінки технічної реалізованості проекту.

Таким чином, описана сервіс дозволяє значно скоротити час, необхідний на попередню обробку заявки, формалізує логіку прийняття рішень та створює основу для подальшої автоматизації та машинного навчання на основі накопичених даних.

4.2 Ролі користувачів та їх функціональність

Для забезпечення максимальної ефективності роботи IT-сервісу було детально визначено основні ролі користувачів, які безпосередньо взаємодіють із сервісом. Кожна роль має унікальний набір функцій і відповідальностей, що оптимізують робочий процес і гарантують якісну

обробку пропозицій від початкового прийому до остаточного рішення. Такий підхід сприяє не лише підвищенню продуктивності, а й забезпеченню чіткого розподілу завдань між учасниками процесу.

4.2.1 Користувачі сервісу

Основним користувачем сервісу є лідогенератор – спеціаліст, який виконує ключову роль у первинному зборі та обробці інформації про потенційні IT-проекти. Його завдання полягає у внесенні початкових даних про пропозиції, отриманих, наприклад, із фриланс-бірж типу Upwork, а також у подальшому контролі за статусом обробки цих пропозицій.

Інтерфейс для лідогенератора максимально спрощений і орієнтований на швидкий і зручний ввід інформації. Лідогенератор має змогу:

- вводити назву проекту, опис, бюджет, терміни виконання, технічний стек, доступність технічних працівників;
- отримувати від сервісу звіти з оцінки технічної реалізованості проєктів;
- переглядати історію опрацьованих пропозицій та статуси їх опрацювання;
- приймати початкові рішення, орієнтуючись на звіт сервісу.

Завдяки цьому користувачеві делегується рутинна частина роботи, що значно підвищує швидкість обробки заявок і знижує навантаження на команду.

4.2.2 Технічний експерт

Наступною важливою роллю є технічний експерт – фахівець, який виконує функції детальної перевірки й уточнення оцінок, сформованих автоматизованим сервісом. Його компетенція дає можливість перевірити технічні аспекти проектів, врахувати специфіку технологій, складність завдань і реальні можливості команди розробників.

Технічний експерт має розширений інтерфейс, що надає:

- доступ до детальних аналітичних даних про проекти;
- можливість коригувати або доповнювати оцінки, внесені сервісом;
- інструменти для коментування пропозицій і внесення зауважень;
- засоби для остаточного підтвердження або відхилення звіту, з урахуванням технічної експертизи.

Цей рівень перевірки забезпечує більш точний і професійний відбір пропозицій, що підвищує якість прийнятих рішень і сприяє зменшенню ризиків невдалих проектів.

4.2.3 Адаптація інтерфейсів під ролі користувачів

Задля підвищення зручності та продуктивності взаємодії кожного користувача із сервісом, інтерфейс сервісу розроблено з урахуванням специфіки виконуваних ним функцій. Лідогенератор отримує максимально спрощений та інтуїтивно зрозумілий інтерфейс для швидкого введення даних, експерт – розгорнуті інструменти для детального аналізу та корекції, а адміністратор – комплексний набір засобів для керування внутрішніми ресурсами сервісу.

Такий поділ не лише полегшує роботу кожного користувача, а й запобігає помилкам, підвищує безпеку і забезпечує цілісність даних [13].

4.2.4 Аналіз потоків даних

У процесі проектування інформаційних систем важливим етапом є аналіз потоків даних, що відображає, як інформація переміщується між елементами сервісу, звідки вона надходить, як обробляється і куди передається. Один із найпоширеніших способів візуалізації цього процесу є діаграма потоків даних (DFD – Data Flow Diagram) [2]. Такі діаграми наочно демонструють, як інформаційні об’єкти переміщуються в межах сервісу, показуючи як зовнішні джерела, так і внутрішні обробники, що трансформують ці дані. На відміну від блок-схем алгоритмів або діаграм класів, DFD не фокусується на логіці виконання або структурах, а саме на обробці інформації, тим самим дозволяючи краще зрозуміти, як і з чим працює сервіс на рівні обміну даними (рисунок 4.1).

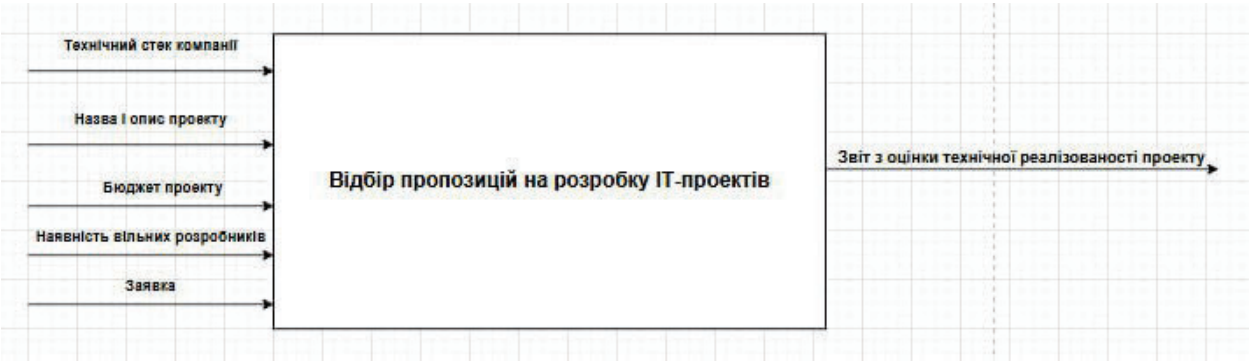

Рисунок 4.1 – Контекстна DFD– діаграма IT-сервісу відбору пропозицій на розробку IT-проектів

Перша діаграма демонструє загальний процес, який охоплює основну логіку функціонування сервісу. Вхідними даними є:

- технічний стек компанії – перелік технологій, з якими працює команда розробників;
- назва і опис проекту – коротка характеристика поставленого завдання;

– бюджет проекту – гранична сума, що може бути витрачена на реалізацію;

– наявність вільних розробників – ресурсний стан команди на момент подання заявки;

– заявка – безпосередній запит на оцінку технічної реалізованості проекту;

– термін виконання – термін, за який треба виконати пропозицію.

Ці дані надходять до модуля «Відбір пропозицій на розробку IT-проектів», який виконує комплексну оцінку. Результатом обробки є звіт з оцінки технічної реалізованості проекту. Такий підхід дозволяє автоматизувати перевірку проектної заявки на відповідність ресурсам та технічним можливостям компанії.

У рамках сервісу виконуються три функції:

– прийом та обробка заявок;

– оцінювання технічної реалізованості проекту;

– формування вихідного документа (рисунок 4.2).

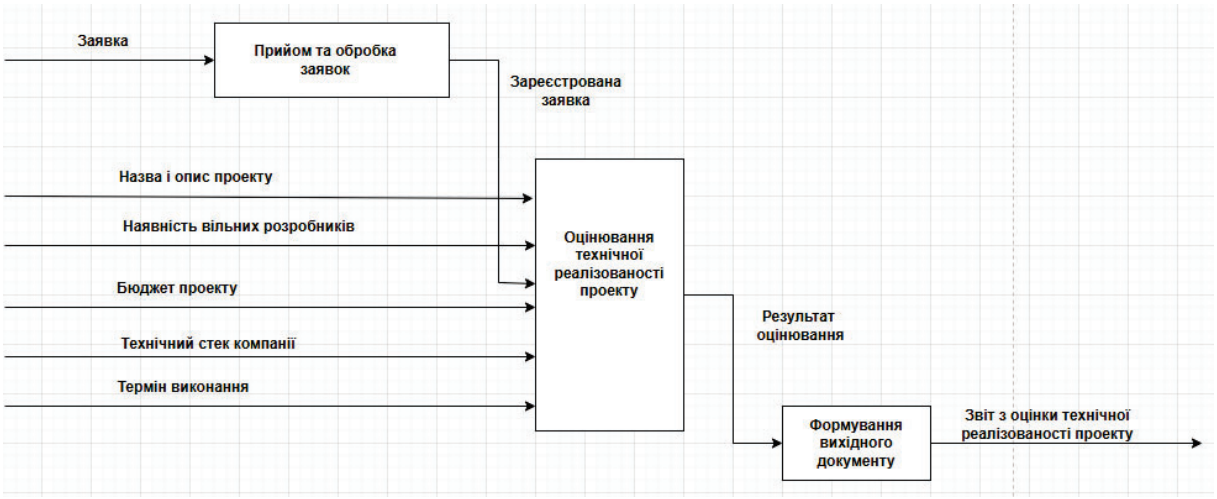

Рисунок 4.2 – DFD - діаграма декомпозиції першого рівня IT-сервісу відбору пропозицій на розробку IT-проектів

Друга діаграма деталізує процес обробки заявки в межах сервісу та

відображає взаємодію функцій.

Прийом та обробка заявок. Початковий етап передбачає реєстрацію заявки в сервісі та її первинну валідацію. Вхідним параметром є сама заявка. Після обробки формується підтвердження замовлення, яке надходить на наступну функцію.

Оцінювання технічної реалізованості проекту. На цьому етапі сервіс отримує усі релевантні параметри, зокрема:

- назва та опис проекту;
- наявність вільних розробників;
- бюджет проекту;
- технічний стек компанії;
- термін виконання.

На основі цих даних відбувається порівняльний аналіз показників запропонованого проекту з показниками компанії (цілі, очікувані результати, обсяги бюджету, терміни та технічну складність). У результаті отримаємо оцінку технічної реалізованості проекту.

Формування вихідного документа. Завершальним етапом є формування звіту, який відображає рівень реалізованості проекту в межах наявних ресурсів і технічного потенціалу. Звіт є фінальним документом, який передається користувачу сервісу – лідогенератору.

Представлені діаграми ілюструють логічну структуру функціонування сервісу оцінювання проектів в ІТ-компанії. Увесь сервіс побудований на поетапному зборі, перевірці та обробці інформації, що дозволяє автоматизувати процес прийняття рішень, зменшити ризики перевантаження команди та забезпечити відповідність проектів технічним можливостям компанії.

Такий підхід забезпечує прозорість, обґрунтованість та ефективність у виборі оптимальних ІТ-проектів для реалізації, що відповідає сучасним вимогам до інформаційних систем керування проектами [12].

Отже, після аналізу архітектури об’єкта розробки на рівні функцій,

можна зробити висновок, що запропонований IT-сервіс є ефективним інструментом автоматизації первинної обробки IT-пропозицій. Завдяки чіткому визначенню вимог, поділу сервісу на функціональні компоненти та реалізації сучасної клієнт-серверної архітектури, досягається висока гнучкість, масштабованість і точність прийняття рішень. Інтеграція інтелектуальних алгоритмів для оцінки релевантності, бюджету та строків реалізації дозволяє суттєво скоротити витрати часу на попередній аналіз і підвищити ефективність роботи лідогенераторів і технічних експертів. Таким чином, сервіс закладає надійне підґрунтя для подальшої оптимізації процесів бізнес-аналітики та розширення функціональності в напрямку прогнозування та машинного навчання.

5 РОЗРОБКА Й ОБҐРУНТУВАННЯ ЕЛЕМЕНТІВ ІНФОРМАЦІЙНОЇ ЗАБЕЗПЕЧУЮЧОЇ СИСТЕМИ ІТ-СЕРВІСУ

5.1 Проектування бази даних

Проектування бази даних є одним із ключових етапів при розробці будь-якої інформаційної системи. У контексті сервісу «Відбір пропозицій на розробку ІТ-проектів» це має вирішальне значення, адже ефективність функціонування системи залежить від швидкого та точного доступу до даних, на основі яких приймаються управлінські рішення. Правильна організація структури бази даних забезпечує не лише оптимальну продуктивність, але й високу масштабованість та безпеку платформи.

Першочерговим завданням при створенні бази даних є ідентифікація основних сутностей, що відображають логіку предметної області. У рамках сервісу можна виділити наступні ключові сутності: користувачі (users), пропозиції (proposals), проекти (projects), теги (tags), етапи розгляду (stages), а також оціночні критерії (criteria). Зв’язки між цими сутностями відображають динаміку відбору проектів, починаючи з моменту подання заявки до остаточного затвердження чи відхилення (рисунок 5.1).

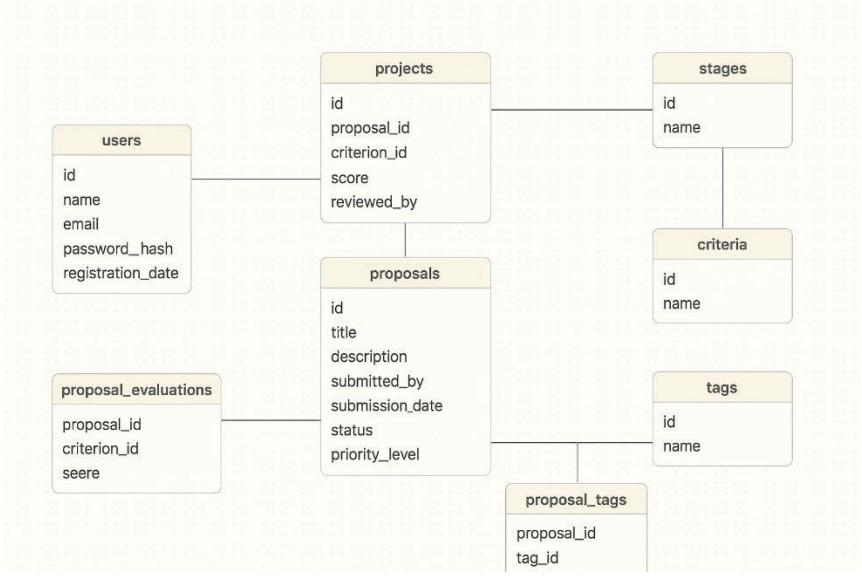

Рисунок 5.1 – Структура бази даних

Кожна сутність має чітко визначений набір атрибутів. Наприклад, таблиця users включає поля id, name, email, role, password_hash, registration_date. Така структура дозволяє ідентифікувати користувачів, розрізняючи адміністраторів, експертів та авторів заявок. Сутність proposals містить атрибути id, title, description, submitted_by, submission_date, status, priority_level, що відображають зміст заявки та її стан.

З метою забезпечення нормалізації структури бази даних було реалізовано 3-тю нормальну форму (3НФ). Це дозволяє уникнути надмірності даних і забезпечити логічну цілісність усього сервісу. Крім того, для оптимізації запитів передбачено створення індексів для полів submission_date, status, priority_level, які є найбільш частими фільтрами при пошуку та сортуванні пропозицій.

У якості системи керування базами даних було обрано PostgreSQL, яка підтримує складні реляційні запити, транзакційність, масштабованість і дозволяє реалізовувати функції з високою продуктивністю. PostgreSQL також має вбудовані засоби для реалізації контролю доступу, що особливо важливо в умовах обробки конфіденційної інформації.

Таким чином, спроектована база даних забезпечує повноцінну підтримку усіх бізнес-процесів, пов’язаних з аналізом і відбором заявок, дозволяє ефективно зберігати, обробляти і аналізувати дані та створює міцний фундамент для розвитку інформаційної системи компанії в майбутньому.

5.2 Реалізація бази даних

У процесі створення IT-сервісу відбору пропозицій на розробку IT-проектів важливою складовою є розробка правильної логічної та фізичної структури бази даних. Ця структура має забезпечувати не лише збереження

та захист інформації, а й її швидку обробку, можливість фільтрації, ранжування й аналітики даних, пов’язаних із IT-проектами, пропозиціями від розробників, критеріями оцінювання та етапами розробки.

Як основну систему керування базами даних обрано PostgreSQL – надійну об’єктно-реляційну СУБД, що дозволяє ефективно працювати з великим обсягом пов’язаних між собою даних, забезпечуючи високий рівень узгодженості та підтримку складних зв’язків.

Архітектура бази даних сформована за допомогою ORM Sequelize, що дозволяє реалізувати підхід Code First – створення структури бази даних із коду, а не навпаки. У результаті було спроектовано наступні основні таблиці: users, projects, stages, criteria, proposals, proposal_evaluation, tags, proposal_tags. Далі детально розглянемо кожну з них.

5.2.1 Таблиця users

Таблиця users зберігає дані про всіх користувачів сервісу. Кожен запис у цій таблиці відповідає унікальному користувачу, який може бути або замовником проекту, або розробником, що подає пропозиції (рисунок 5.2).

```
CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    username VARCHAR(50) NOT NULL UNIQUE,
    email VARCHAR(100) NOT NULL UNIQUE,
    password_hash TEXT NOT NULL,
    role VARCHAR(20) CHECK (role IN ('sales', 'developer', 'leadgen')) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Рисунок 5.2 – Вигляд таблиці користувача

Основні поля:

– id – унікальний ідентифікатор користувача (primary key,

автоінкремент);

- username – унікальне ім’я користувача;
- email – електронна адреса, також унікальна;
- password_hash – хешований пароль (для безпеки);
- role – тип користувача (customer, developer, admin);
- created_at, updated_at – час створення й останнього оновлення запису.

Функціональне значення: ця таблиця є основою для реалізації автентифікації та авторизації, а також зв’язується з іншими таблицями, зокрема projects, proposals, proposal_evaluation.

5.2.2 Таблиця projects

Таблиця projects містить опис усіх IT-проектів, які створюються замовниками (рисунок 5.3).

```
CREATE TABLE projects (
    id SERIAL PRIMARY KEY,
    user_id INTEGER NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    title VARCHAR(255) NOT NULL,
    description TEXT,
    budget NUMERIC(12, 2),
    deadline DATE,
    status VARCHAR(20) CHECK (status IN ('open', 'in_progress', 'completed', 'closed')) DEFAULT '
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Рисунок 5.3 – Вигляд таблиці проектів

Основні поля:

- id – унікальний ідентифікатор проекту;
- user_id – зовнішній ключ до таблиці users, що вказує, хто створив проект;
- title – назва проекту;

- description – розгорнутий опис задачі;
- budget – очікуваний бюджет;
- deadline – кінцева дата завершення;
- status – поточний статус (open, closed, in_progress, completed);
- created_at, updated_at.

Зв’язки: один користувач може створити багато проектів, але кожен проект належить одному користувачу. Також проект пов'язаний із таблицею stages, tags та proposals.

5.2.3 Таблиця stages

Таблиця stages реалізує концепцію етапів розробки в межах одного проекту. Це дозволяє ділити складний IT-проект на частини з різними термінами виконання, відповідальними й бюджетами (рисунок 5.4).

```
CREATE TABLE stages (
    id SERIAL PRIMARY KEY,
    project_id INTEGER NOT NULL REFERENCES projects(id) ON DELETE CASCADE,
    title VARCHAR(255) NOT NULL,
    description TEXT,
    deadline DATE,
    status VARCHAR(20) CHECK (status IN ('planned', 'in_progress', 'completed')) DEFAULT 'planned
);
```

Рисунок 5.4 – Вигляд таблиці стадій обробки заявки

Основні поля:

- id – унікальний ідентифікатор етапу;
- project_id – зовнішній ключ до таблиці projects;
- title – назва етапу;
- description – опис функціоналу або задач;
- deadline – строк завершення етапу;
- status – поточний стан етапу (planned, in_progress, completed).

Зв’язки: один проект може мати кілька етапів, але кожен етап належить лише одному проекту.

5.2.4 Таблиця criteria

Таблиця criteria описує критерії оцінювання пропозицій розробників. Кожен критерій може застосовуватись до різних проектів, залежно від пріоритетів замовника (рисунок 5.5).

```
CREATE TABLE criteria (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL UNIQUE,
    description TEXT,
    weight NUMERIC(5, 2) CHECK (weight >= 0 AND weight <= 1)
);
```

Рисунок 5.5 – Вигляд таблиці критеріїв

Основні поля:

- id – унікальний ідентифікатор критерію;
- name – назва критерію (наприклад, "вартість", "досвід", "якість портфоліо");
- description – детальний опис, як цей критерій оцінюється;
- weight – числове значення, що визначає важливість критерію (0–1 або у відсотках).

Призначення: дозволяє реалізувати гнучку багатофакторну модель оцінювання, яка комбінує об’єктивні й суб’єктивні показники.

5.2.5 Таблиця proposals

proposals – це таблиця, яка зберігає усі подані розробниками пропозиції до відкритих проектів (рисунок 5.6).

```
CREATE TABLE proposals (
    id SERIAL PRIMARY KEY,
    developer_id INTEGER NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    project_id INTEGER NOT NULL REFERENCES projects(id) ON DELETE CASCADE,
    description TEXT,
    price NUMERIC(12, 2),
    duration INTEGER CHECK (duration > 0),
    submitted_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Рисунок 5.6 – Вигляд таблиці заявок

Основні поля:

- id – унікальний ідентифікатор пропозиції;
- developer_id – зовнішній ключ до таблиці users;
- project_id – зовнішній ключ до таблиці projects;
- description – короткий опис, як виконавець планує реалізувати проект;
- price – запропонована ціна;
- duration – тривалість реалізації (у днях);
- submitted_at – дата подання пропозиції.

Зв’язки: одна пропозиція належить до одного проекту, але один користувач може подавати багато пропозицій до різних проектів.

5.2.6 Таблиця proposal_evaluation

Таблиця proposal_evaluation фіксує оцінки пропозицій відповідно до критеріїв, зазначених у criteria. Це реалізація моделі N:M:M – один критерій

застосовується до однієї пропозиції в межах одного проекту (рисунок 5.7).

```
CREATE TABLE proposal_evaluation (
    id SERIAL PRIMARY KEY,
    proposal_id INTEGER NOT NULL REFERENCES proposals(id) ON DELETE CASCADE,
    criterion_id INTEGER NOT NULL REFERENCES criteria(id) ON DELETE CASCADE,
    score NUMERIC(4, 2) CHECK (score >= 0 AND score <= 10),
    UNIQUE (proposal_id, criterion_id)
);
```

Рисунок 5.7 – Вигляд таблиці фінального рішення

Основні поля:

- id – унікальний ідентифікатор оцінювання;
- proposal_id – зовнішній ключ до таблиці proposals;
- criterion_id – зовнішній ключ до таблиці criteria;
- score – числова оцінка (0–10 або за шкалою 100).

Функціональність: дозволяє автоматизовано обчислювати загальну оцінку пропозиції з урахуванням вагових коефіцієнтів.

5.2.7 Таблиця tags

Таблиця tags містить ключові слова або технології, що використовуються для фільтрації та категоризації проектів (рисунок 5.8).

```
CREATE TABLE tags (
    id SERIAL PRIMARY KEY,
    name VARCHAR(50) NOT NULL UNIQUE
);
```

Рисунок 5.8 – Вигляд таблиці тегів

Основні поля:

- id – унікальний ідентифікатор тега;

– name – назва тега (наприклад, React, Node.js, UI/UX, Мобільна розробка).

5.2.8 Таблиця proposal_tags

Зв’язувальна таблиця proposal_tags реалізує зв’язок "багато до багатьох" між пропозиціями (proposals) та тегами (tags). Наприклад, пропозиція може містити кілька тегів, що описують стек технологій, якими володіє розробник (рисунок 5.9).

```
CREATE TABLE proposal_tags (
    id SERIAL PRIMARY KEY,
    proposal_id INTEGER NOT NULL REFERENCES proposals(id) ON DELETE CASCADE,
    tag_id INTEGER NOT NULL REFERENCES tags(id) ON DELETE CASCADE,
    UNIQUE (proposal_id, tag_id)
);
```

Рисунок 5.9 – Вигляд зв’язної таблиці тегів та заявок

Основні поля:

– id – ідентифікатор запису;

– proposal_id – зовнішній ключ до proposals;

– tag_id – зовнішній ключ до tags;

Загальна структура взаємозв’язків:

– один користувач може створити багато проектів (users → projects);

– один проект може мати багато етапів (projects → stages);

– один проект отримує багато пропозицій (projects → proposals);

– одна пропозиція може бути оцінена за багатьма критеріями (proposals ↔ criteria ↔ proposal_evaluation);

– одна пропозиція може мати багато тегів (proposals ↔ tags ↔ proposal_tags).

Також для підвищення продуктивності виконуються такі дії (рисунок

5.10).

```
CREATE INDEX idx_projects_user_id ON projects(user_id);
CREATE INDEX idx_proposals_project_id ON proposals(project_id);
CREATE INDEX idx_proposals_developer_id ON proposals(developer_id);
CREATE INDEX idx_proposal_evaluation_proposal_id ON proposal_evaluation(proposal_id);
CREATE INDEX idx_proposal_evaluation_criterion_id ON proposal_evaluation(criterion_id);
```

Рисунок 5.10 – Створення індексацій для таблиць

Розроблена структура бази даних забезпечує високий рівень нормалізації, що мінімізує дублювання інформації та полегшує масштабування сервісу. Кожна таблиця виконує чітко визначену функцію в логіці роботи сервісу, дозволяючи реалізувати гнучкий механізм взаємодії між користувачами, проектами та розробниками. Окрему увагу приділено реалізації системи оцінювання, яка дозволяє формувати об’єктивні рейтинги пропозицій на основі вагових коефіцієнтів.

6 РОЗРОБКА Й ОБҐРУНТУВАННЯ ЕЛЕМЕНТІВ ПРОГРАМНОЇ ЗАБЕЗПЕЧУЮЧОЇ СИСТЕМИ ІТ-СЕРВІСУ

6.1 Обґрунтування застосування технологій штучного інтелекту для вибору пропозицій на розробку ІТ-проектів

У процесі формування портфеля проектів для реалізації надзвичайно важливим є питання об'єктивної перевірки кожної поданої пропозиції. На практиці ІТ-компанії часто стикаються з необхідністю оцінити велику кількість заявок, кожна з яких має свої унікальні параметри: цілі, очікувані результати, обсяги бюджету, терміни та технічну складність. У традиційній моделі ця задача зазвичай вирішується за допомогою експертного аналізу, однак подібний підхід має суттєві обмеження. По-перше, він передбачає витрату значного обсягу людських ресурсів. По-друге, процес аналізу сильно залежить від суб'єктивного досвіду та бачення конкретного фахівця. По-третє, при зростанні кількості заявок збільшується ймовірність помилок, пропусків або неточностей, що безпосередньо впливає на якість прийнятих рішень.

У зв’язку з цим особливої актуальності набуває використання інструментів штучного інтелекту, які здатні здійснювати автоматизовану, масштабовану та адаптивну валідацію проектних пропозицій. Інтелектуальні алгоритми не лише зменшують навантаження на експертів, а й забезпечують більшу послідовність у прийнятті рішень, зменшуючи вплив людського фактору на процес оцінювання [5].

ШІ-моделі мають ряд важливих переваг, які ілюструють доцільність їхнього використання в межах розробки даного сервісу:

− прогнозування результативності. На основі аналізу значного масиву попередніх даних (включаючи характеристики успішних і неуспішних проектів, їхню тематику, складність, часові та фінансові рамки) сервіс здатний з великою ймовірністю передбачити потенційний результат нової

заявки;

− класифікація заявок за ризиком та пріоритетом. ШІ-моделі, зокрема ті, що базуються на методі градієнтному бустингу, деревоподібних алгоритмах чи нейронних мережах, можуть автоматично розподіляти проекти за категоріями, визначаючи, наскільки вони відповідають поточній стратегії компанії;

− самонавчання сервісу. Завдяки технологіям машинного навчання моделі покращують точність прийнятих рішень у міру накопичення нових прикладів. Це означає, що з часом сервіс стає ще більш точним та адаптованою до специфіки конкретної організації;

− швидкість та масштабність. Автоматизована модель здатна обробити сотні заявок протягом хвилин, що особливо актуально у великих компаніях або під час кампаній з активного залучення клієнтів;

− зниження впливу людського фактору. Упередженість, втома чи інші психологічні аспекти, які можуть впливати на оцінювача, усуваються завдяки використанню математично обґрунтованих моделей.

У межах цього дослідження передбачається реалізація окремого модуля ШІ-валідації, який буде включати:

− обробку природної мови (NLP) для глибокого аналізу текстового опису заявки, виявлення ключових слів, концептуальних категорій та контексту;

− моделі багатофакторної регресії, що дозволять визначити відповідність заявки певним критеріям (наприклад, бюджету, термінам, технічному стеку) та оцінити її релевантність;

− алгоритми скорингу, які оцінюватимуть кожен проект за певними балами на основі тренувальної вибірки з попередніх реалізованих задач;

− сервіс виявлення аномалій, що допоможуть виявити заявки з підозрілими або нетиповими параметрами (наприклад, надмірно короткі строки реалізації при високому бюджеті тощо).

Таким чином, упровадження модуля ШІ в процес валідації проектних заявок дозволить не лише оптимізувати витрати на обробку даних, а й підвищити об'єктивність та аналітичну глибину прийнятих рішень. Це сприятиме створенню надійного, прозорого й ефективного механізму відбору проектів, що відповідають стратегічним цілям компанії.

6.2 Обґрунтування вибору React.js для клієнтської частини та Node.js, Postgre для серверної частини

У процесі проектування сучасних IT-сервісів одним із найважливіших етапів є формування технологічного стеку, що забезпечить ефективну розробку, легкість масштабування, швидке реагування сревісу на запити користувачів та відповідність актуальним стандартам у сфері безпеки й зручності користування. Для реалізації сервісу відбору пропозицій на розробку IT-проектів було обрано фул-стек архітектуру, основою якої є React.js для клієнтської частини, а також Node.js у поєднанні з Postgre для серверної логіки та зберігання даних. Такий підхід є виправданим як з технічної, так і з економічної точок зору, оскільки дозволяє створити надійний, швидкодіючий та адаптивний веб-застосунок.

6.2.1 React.js як основа інтерфейсної частини

React.js – це потужна JavaScript-бібліотека, яка розроблена компанією Meta з метою створення динамічних, зручних та масштабованих користувацьких інтерфейсів [7]. Основною перевагою цієї бібліотеки є її компонентно-орієнтована структура, що дає змогу будувати застосунок із

незалежних блоків (компонентів), які можна багаторазово використовувати, комбінувати та оновлювати без необхідності переробки всього сервісу.

Суттєвою технічною перевагою React.js є використання віртуального DOM – внутрішньої репрезентації структури сторінки, що дає змогу швидко відслідковувати зміни та вносити лише необхідні оновлення у реальний DOM, що, у свою чергу, значно покращує швидкодію інтерфейсу. Крім того, React має широку підтримку з боку спільноти розробників, що забезпечує доступ до великої кількості готових бібліотек, інструментів для тестування, розширення функціоналу, а також регулярні оновлення безпеки.

Ще однією важливою характеристикою React є його гнучкість – технологія легко інтегрується з RESTful API, GraphQL, WebSockets та іншими сучасними інструментами, що дозволяє реалізувати складну інтерактивну логіку без втрати стабільності. У контексті розробки сервісу, орієнтованого на роботу з великою кількістю форм, таблиць і динамічного контенту, саме React дозволяє створити зручний інтерфейс для подачі заявок, перегляду статусу проекту та управління пропозиціями.

6.2.2 Node.js як платформа для серверної логіки

Node.js – це середовище виконання JavaScript-прикладного коду на сервері, що дозволяє створювати масштабовані веб-сервіси з високою продуктивністю [7]. Основна перевага Node.js – неблокуюча модель вводу/виводу (event-driven architecture), яка дає змогу обробляти велику кількість одночасних запитів без зниження швидкодії, що критично важливо для сервісів, які опрацьовують численні запити від користувачів в режимі реального часу.

Оскільки і клієнтська, і серверна частина застосунку розробляються на одній мові – JavaScript [9], це значно спрощує процес розробки,

тестування та супроводу проекту. Такий підхід мінімізує бар'єри між клієнтською і серверною командами, а також дає змогу використовувати єдиний стек технологій, що спрощує передачу даних, реалізацію логіки взаємодії, інтеграцію API та розробку документації.

Завдяки багатій екосистемі npm-пакетів Node.js дозволяє швидко реалізовувати типові задачі – від авторизації користувачів до побудови REST API. Простота налаштування серверного середовища, висока гнучкість та швидкість розгортання додатково сприяють зниженню витрат на запуск і підтримку проекту.

6.2.3 PostgreSQL як система зберігання даних

PostgreSQL є об'єктно-реляційною системою керування базами даних (СКБД), яка відома своєю стабільністю, відповідністю стандартам SQL та широкими можливостями для розробників [12]. У контексті реалізації IT-сервісу відбору пропозицій на розробку IT-проєктів, PostgreSQL виступає основною платформою для надійного зберігання та обробки структурованих даних.

На відміну від NoSQL-рішень, PostgreSQL передбачає чітке визначення схеми даних, що дозволяє забезпечити цілісність, контроль типів і зв'язків між таблицями. Така структура ідеально підходить для побудови інформаційних систем із чіткими бізнес-правилами, складною логікою і суворими вимогами до цілісності записів. Зокрема, застосування зовнішніх ключів (foreign keys), обмежень цілісності (constraints) та індексів дозволяє підтримувати високу якість і узгодженість даних у базі.

PostgreSQL також підтримує потужні засоби масштабування – як вертикального, так і горизонтального, з можливістю налаштування реплікації, шардінгу та балансування навантаження. Це дозволяє базі даних

ефективно обслуговувати значну кількість одночасних запитів при високих навантаженнях. Крім того, PostgreSQL має підтримку JSON та JSONB типів даних, що надає часткову гнучкість, подібну до NoSQL-рішень, у випадках, коли потрібно зберігати напівструктуровані або змінні за формою дані.

У поєднанні з Node.js PostgreSQL забезпечує ефективний і безпечний обмін даними через API. Для взаємодії між серверною частиною на Node.js та PostgreSQL використовуються популярні бібліотеки, такі як pg або ORM-фреймворки типу Sequelize чи Prisma. Це значно спрощує розробку, підтримку та масштабування програми, забезпечуючи прозоре керування запитами, транзакціями та обробкою помилок.

PostgreSQL також має широкий спектр додаткових можливостей, зокрема повнотекстовий пошук, роботу з геоданими (через PostGIS), розширення для аналітики й статистики, що робить її універсальним інструментом для складних і масштабованих проєктів.

Таким чином, використання PostgreSQL у складі обраного стеку технологій (React.js + Node.js + PostgreSQL) створює міцну архітектурну основу для розробки масштабованого, безпечного та надійного веб-сервісу. Така конфігурація дозволяє ефективно реалізувати і адміністративний інтерфейс для керування даними, і зручний користувацький інтерфейс, а також гарантує простоту підтримки й можливість подальшого розширення функціоналу без необхідності кардинальних змін у структурі застосунку.

6.3 API-інтерфейс

У сервісі реалізовано RESTful API-інтерфейс, який дозволяє клієнтським застосункам або стороннім сервісам безпечно взаємодіяти з функціоналом серверної частини. API побудовано на принципах чіткого поділу обов’язків: кожен маршрут відповідає за окрему дію – реєстрацію,

автентифікацію або отримання профілю користувача. Обмін інформацією відбувається у форматі JSON, що забезпечує простоту інтеграції та високу продуктивність. Кожен запит має чітко описану структуру, очікувану відповідь та обов’язкові параметри.

Основні запити не потребують авторизації (реєстрація та логін), однак запити до захищених маршрутів, зокрема до профілю користувача, вимагають передачі дійсного JWT-токена у заголовках запиту. Такий підхід гарантує безпечний обмін даними між клієнтом та сервером і дозволяє реалізовувати політику доступу відповідно до ролей користувача.

6.3.1 POST /api/register

Завданням цього АПІ є забезпечення процесу реєстрації нового користувача на платформі. Реєстрація передбачає створення унікального облікового запису, який дозволяє користувачу отримати доступ до функціональних можливостей сервісу відповідно до наданих прав і ролей (рисунок 6.3).. Під час реєстрації від користувача збираються необхідні дані, які проходять перевірку на коректність і відповідність вимогам безпеки (рисунок 6.4). Цей процес є ключовим для встановлення персоналізованого середовища взаємодії та забезпечення подальшої автентифікації й авторизації в системі. Таким чином, реєстрація нового користувача виступає першочерговим кроком для початку ефективної роботи із сервісом.

Тіло запиту буде виглядати наступним чином:

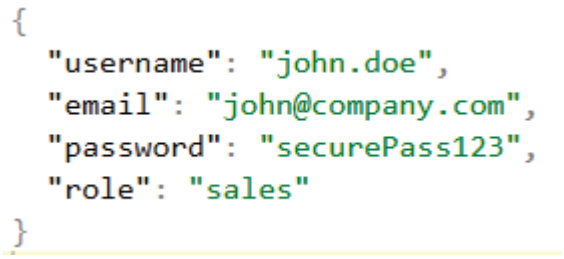

Рисунок 6.3 – Тіло запиту на реєстрацію користувача

Як успішна відповідь від серверу на успішну реєстрацію можна буде побачити наступне:

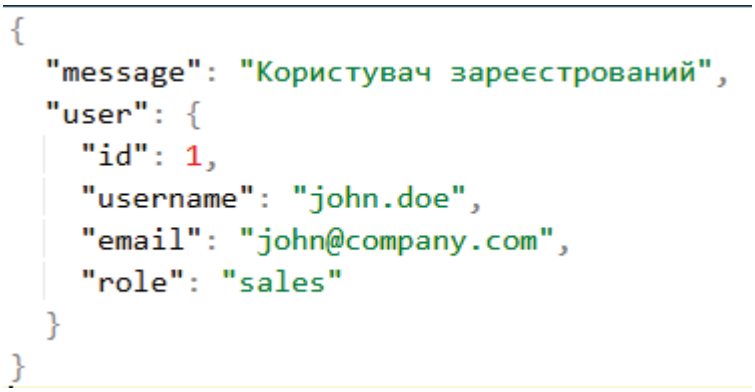

Рисунок 6.4 – Відповідь від сервера на успішну реєстрації

6.3.2 POST /api/login

Цей інтерфейс відповідає за процес авторизації користувача в сервісі та видачу йому JWT-токена (рисунок 6.5). Авторизація полягає у перевірці введених користувачем облікових даних для підтвердження його особи та надання доступу до захищених ресурсів платформи. Після успішної аутентифікації система генерує JWT-токен, який служить підтвердженням права користувача на подальшу взаємодію з сервісом без необхідності повторної авторизації на кожному кроці (рисунок 6.6). Це підвищує зручність використання та забезпечує безпечний спосіб управління сесіями користувачів.

Тіло запиту буде виглядати так:

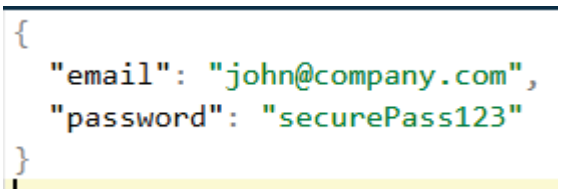

Рисунок 6.5 – Тіло запиту на авторизацію користувача

Відповідь на авторизацію від сервера:

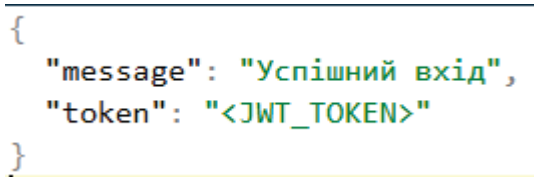

Рисунок 6.6 – Відповідь серверу на запит авторизації

6.3.3 GET /api/profile

Цей функціональний компонент відповідає за надання інформації про поточного авторизованого користувача на основі переданого токена доступу. Його головне призначення – забезпечити можливість отримання актуальних даних профілю, таких як ім’я, роль у системі, права доступу або інші персоналізовані параметри. Запит доступний лише для користувачів із дійсною авторизацією, що гарантує безпеку обробки персональних відомостей. Такий підхід дозволяє динамічно адаптувати інтерфейс і функціонал сервісу відповідно до ролі користувача.

До цього запиту також існує окремий заголовок, який можна було отримати від серверу під час авторизації і треба передати до серверу зараз, щоб сервер зміг зрозуміти, що юзер авторизований і надавати доступ до даних безпечно: Authorization: Bearer <JWT_TOKEN> (рисунок 6.7).

Успішна відповідь від серверу:

Рисунок 6.7 – Відповідь серверу на запит даних користувача

6.4 Реалізація на Node.js

Реалізація функціоналу серверної частини виконана з використанням популярного технологічного стеку Node.js + Express.js. Така архітектура дозволяє створювати гнучкі RESTful-сервіси з високою масштабованістю та низьким порогом входу для розробників. Кожен компонент розділено на окремі модулі: сервер, маршрутизація (routes), обробка підключення до бази даних та middleware для захисту.

6.4.1 Підключення до бази PostgreSQL

Для взаємодії з базою даних PostgreSQL використовується бібліотека pg з конфігурацією підключення через змінні середовища. Це дозволяє дотримуватися принципів безпеки та легко адаптувати конфігурацію для продакшн- або тестового середовища (рисунок 6.8).

```
const { Pool } = require('pg');
require('dotenv').config();

const pool = new Pool({
  connectionString: process.env.DATABASE_URL,
});

module.exports = pool;
```

Рисунок 6.8 – Встановлення підключення сервера до бази даних

6.4.2 Основний сервер

Основний сервер ініціалізується з використанням Express. Підключаються middleware cors для обробки запитів з інших доменів та body-parser для зчитування JSON у тілі запиту. Всі маршрути зібрані у файлі routes/auth.js і доступні за шляхом /api (рисунок 6.9).

```
const express = require('express');
const cors = require('cors');
const bodyParser = require('body-parser');
const authRoutes = require('./routes/auth');

const app = express();
app.use(cors());
app.use(bodyParser.json());

app.use('/api', authRoutes);

app.listen(5000, () => console.log('Сервер запущено на порту 5000'));
```

Рисунок 6.9 – Ініціалізація серверу

Завдяки цій ініціалізаці запускається сервер, який забезпечує зв’язок веб та БД частини сервісу.

6.4.3 Роутинг /routes/auth.js

Файл auth.js відповідає за основну логіку реєстрації, логіну та отримання профілю. Усі запити до бази даних обгорнуті у try-catch для обробки винятків, а всі критичні операції (як-от створення користувача або перевірка паролю) супроводжуються валідацією введених даних.

Маршрут /register перевіряє домен email, хешує пароль і створює нового користувача в базі.

Маршрут /login перевіряє email, порівнює хеш пароля і повертає JWT.

Маршрут /profile – захищений middleware-ом і повертає інформацію про користувача.

6.4.4 Middleware auth.js

Цей файл перевіряє наявність токена у заголовках запиту. Якщо токен валідний – витягується payload і прикріплюється до об’єкта req.user, що дозволяє ідентифікувати користувача в наступних функціях маршрутизації (рисунок 6.10).

```
const jwt = require('jsonwebtoken');

function authMiddleware(req, res, next) {
  const token = req.headers.authorization?.split(' ')[1];
  if (!token) return res.status(401).json({ message: 'Токен не надано' });
  try {
    const decoded = jwt.verify(token, process.env.JWT_SECRET);
    req.user = decoded;
    next();
  } catch {
    res.status(403).json({ message: 'Невалідний токен' });
  }
}

module.exports = authMiddleware;
```

Рисунок 6.10 – Реалізація валідації токену

Таким чином, допоміжний функціонал реалізований відповідно до сучасних стандартів безпеки та гнучкості. Він забезпечує повноцінну взаємодію користувача із сервісом – від моменту реєстрації до виконання оцінки IT-проектів через ChatGPT, гарантуючи водночас контроль доступу до конфіденційної інформації.

6.5 Програмна реалізація форми валідації проекту

У контексті розробки сервісу автоматизованої оцінки та відбору IT-проектів, форма валідації проекту є ключовим компонентом, через який здійснюється попередній аналіз отриманих пропозицій. Основне завдання цієї форми – забезпечити збирання, обробку й попередню інтерпретацію даних за участю ChatGPT, який виступає як аналітичний асистент. Вона дає змогу перевірити відповідність проектів технічним можливостям компанії, розрахувати орієнтовний бюджет, оцінити доступність фахівців і передбачити можливість вкластися у терміни виконання.

Реалізація цього модуля базується на покроковому user-flow та чітко структурованих критеріях, що дозволяють автоматизувати процес ухвалення рішень щодо релевантності запропонованих проектів. Особливу увагу в процесі реалізації було приділено зручності взаємодії з користувачем, чіткості інтерфейсу, логіці маршрутизації між етапами та обробці даних, що надходять через API.

6.5.1 Загальний алгоритм роботи форми

Форма валідації реалізується як багатокроковий інтерфейс користувача. Кожен етап форми виконує окрему логічну функцію та передбачає взаємодію з ChatGPT, що аналізує введену інформацію в режимі реального часу. Завдяки інтеграції з backend-частиною, користувач отримує миттєвий зворотний зв'язок та звіти щодо подальших дій.

Етапи форми:

− введення назви та опису пропозиції. Користувач вводить загальну інформацію про проект. Ці дані передаються на сервер, де GPT-асистент визначає, до якого технічного напрямку належить проект, та чи відповідає він технічним можливостям компанії;

− введення бюджету. На цьому кроці лідогенератор вказує передбачувану вартість проекту. Сервіс порівнює цей бюджет із внутрішніми ставками компанії. Якщо бюджет невідповідний – сервіс надає можливість вказати, що бюджет може бути переглянутий

− оцінка доступності спеціалістів. GPT аналізує графік роботи розробників у базі даних, щоб зрозуміти, чи зможе команда взятися за проект у найближчі терміни. Якщо ні – користувач обирає: чекати, наймати нового працівника або відмовитися від участі;

− оцінка термінів виконання. Якщо в пропозиції вказано термін

завершення, система оцінює його реалістичність. У випадку збігу – користувач продовжує роботу. Інакше – може сформувати застереження для команди;

– генерація естимації. Завершальний етап – створення автоматичного прогнозу обсягу робіт, який проходить технічну перевірку спеціалістом і зберігається в системі як результат попередньої оцінки.

6.5.2 Програмна реалізація: клієнтська частина (React.js)

Інтерфейс форми реалізовано з використанням бібліотеки React.js, яка забезпечує ефективну побудову компонентно-орієнтованих UI. Кожен крок форми реалізовано як окремий функціональний компонент, що дозволяє масштабувати, тестувати та легко модифікувати логіку у разі змін.

Організація форми:

Форма побудована за принципом багатокрокового майстра (multi-step form), що дозволяє користувачеві переходити від одного кроку до іншого лише після успішної перевірки введених даних. Для обробки форм використовуються react-hook-form, yup (для валідації) та axios для взаємодії з серверною частиною.

На першому кроці водяться дані про технічний стек за допомогою форми валідації технічного стеку (рисунок 6.11).

```jsx
import React from 'react';
import { useForm, Controller } from 'react-hook-form';
import axios from 'axios';

function StepOne({ onNext }) {
  const { control, handleSubmit } = useForm();

  const onSubmit = async (data) => {
    const response = await axios.post('/api/validate/step1', data);
    if (response.data.match) {
      onNext();
    } else {
      alert('Проєкт не відповідає технічним можливостям компанії.');
    }
  };

  return (
    <form onSubmit={handleSubmit(onSubmit)}>
      <Controller
        name="title"
        control={control}
        defaultValue=""
        render={({ field }) => <input {...field} placeholder="Назва проєкту" required />}
      />
      <Controller
        name="description"
        control={control}
        defaultValue=""
        render={({ field }) => <textarea {...field} placeholder="Опис проєкту" required />}
      />
      <button type="submit">Продовжити</button>
    </form>
  );
}
```

Рисунок 6.11 – Програмний код форми валідації технічного стеку

Цей підхід дозволяє отримати результат аналізу без оновлення сторінки, роблячи взаємодію максимально плавною та ефективною для користувача.

Серверна частина відповідає за аналіз введених користувачем даних, взаємодію з базою даних компанії та виклики до GPT API. У структурі серверної частини є окремі маршрути (routes), що відповідають за обробку кожного етапу форми. Крім того, створено окремі сервіси для взаємодії з GPT, обробки внутрішніх даних та логування процесів (рисунок 6.12).

Підключення ChatGPT та обробка першого кроку:

```
const express = require('express');
const router = express.Router();
const { analyzeProposal } = require('../services/gptService');
const { isStackSupported } = require('../services/internalData');

router.post('/validate/step1', async (req, res) => {
  const { title, description } = req.body;
  const gptResult = await analyzeProposal(title, description);
  const match = await isStackSupported(gptResult.stack);
  res.json({ match, stack: gptResult.stack });
});
```

Рисунок 6.12 – Підключення ChatGPT до IT-сервісу

Взаємодія з GPT.

Спеціальний модуль gptService.js призначений для формування запиту GPT. GPT виконує запит, аналізуючи пропозиції за такими критеріями: назва та опис. Програмний код модуля gptService.js показано на рисунку 6.13.

```
const { Configuration, OpenAIApi } = require('openai');

const configuration = new Configuration({ apiKey: process.env.OPENAI_API_KEY });
const openai = new OpenAIApi(configuration);

async function analyzeProposal(title, description) {
  const prompt = `Проаналізуй проєкт з назвою "${title}" та описом: ${description}. Визнач стек (
  const completion = await openai.createChatCompletion({
    model: 'gpt-4',
    messages: [{ role: 'user', content: prompt }],
  });
  const response = completion.data.choices[0].message.content;
  // логіка вилучення стеку з відповіді
  return { stack: extractStack(response) };
}
```

Рисунок 6.13 – Програмний код модуля gptService.js

Модуль gptService.js, дозволяє динамічно формувати запити до GPT, обробляти їх і формувати звіти для прийняття рішення щодо розробки

проектів.

Таким чином, розроблений модуль з валідацією проекту є прикладом сучасного підходу до автоматизації бізнес-процесів у сфері IT-консалтингу. Він дозволяє швидко й ефективно аналізувати нові запити, проводити первинний технічний аналіз та створювати базову естимацію із залученням штучного інтелекту. Завдяки грамотному розділенню відповідальностей між клієнтською і серверною частинами, а також інтеграції з GPT, форма забезпечує не лише зручність для користувача, а й точність у прийнятті рішень.

Унікальність цього підходу полягає у поєднанні форми на React з потужним обробником на Node.js та розумною логікою ChatGPT, яка виконує аналітичну роботу. Така архітектура є масштабованою, розширюваною та адаптивною до зміни бізнес-вимог у майбутньому.

6.6 Реалізація візуального інтерфейсу

Розробка візуального інтерфейсу є одним із ключових етапів створення сучасного сервісу для автоматизації валідації та оцінки IT-проектів. Саме через інтерфейс користувач взаємодіє з платформою, тож важливо забезпечити зручність, логічну послідовність дій і чіткий зворотний зв’язок. У межах реалізації інтерфейсу для IT-сервісу "Proposal Analyzer" були визначені такі основні екрани:

- форма авторизації користувача;
- покрокова форма валідації пропозиції;
- екран з результатом GPT-аналізу;
- сторінка підтвердження естимації технічним спеціалістом.

Ці інтерфейсні компоненти утворюють єдиний логічний ланцюг, що відповідає всім етапам роботи сервісу – від входу в систему до фінального

рішення щодо доцільності участі компанії у проекті.

6.6.1 Сторінка авторизації

Авторизація є першим кроком взаємодії з платформою. Сторінка виконана у стилі лаконічного корпоративного інтерфейсу та містить поля для введення email-адреси й пароля. Головною особливістю реалізації є перевірка доменної належності – реєстрація можлива лише для корпоративних поштових адрес. Це обмежує доступ сторонніх осіб до внутрішньої сервісу та забезпечує базову безпеку даних (рисунок 6.14).

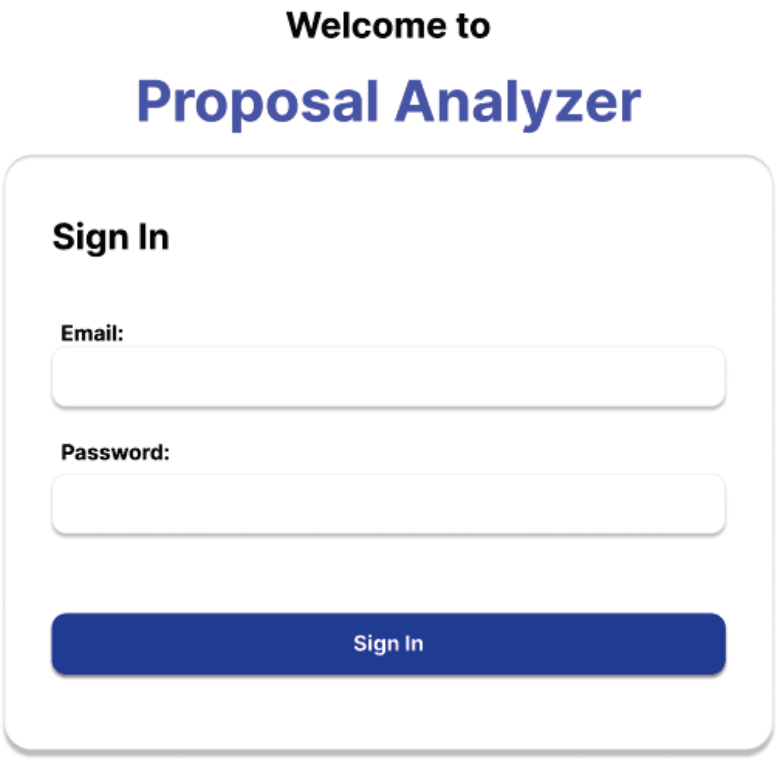

Рисунок 6.14 – Веб-сторінка авторизації

Коли користувач вводить облікові дані, система перевіряє їх і, у разі

успіху, надає доступ до сервісу. У разі помилки у паролі або email'і користувач одразу бачить повідомлення з поясненням проблеми. Крім цього, реалізовано логіку збереження введених даних при помилках, що полегшує повторну спробу входу.

6.6.2 Покрокова форма валідації пропозиції

Основним компонентом візуального інтерфейсу є інтерактивна форма валідації проекту. Вона побудована за принципом multi-step, де кожен етап відповідає за окрему частину аналізу. Інтерфейс створено так, щоб користувач не був перевантажений полями: кожен крок містить мінімум елементів, але забезпечує максимальну інформативність (рисунок 6.15).

Крок 1: Заповнення назви та опису проекту

Після входу користувач потрапляє на перший етап – введення основної інформації про проект. Тут користувач зазначає назву пропозиції та її короткий опис. Після натискання кнопки "Проаналізувати" дані передаються до GPT-асистента. У відповідь чат повертає тип технічного стеку (наприклад, Web, iOS, Android) та інформацію щодо його відповідності технічним можливостям компанії (рисунок 6.16).

Якщо GPT визначає, що проект не відповідає технічному профілю команди, інтерфейс виводить сповіщення та пропонує або змінити опис, або завершити процес.

Крок 2: Введення бюджету

На цьому кроці користувач вводить суму бюджету, запропонованого в пропозиції. Цей параметр надсилається на сервер, де порівнюється із середніми внутрішніми ставками розробників. Якщо бюджет надто низький, з’являється попередження із коментарем GPT та чекбокс "Ігнорувати обмеження", що дозволяє рухатись далі в разі особливо цінних проектів.

Таке рішення дає змогу уникнути автоматичного відсіву перспективних, але недостатньо оцінених замовником проектів.

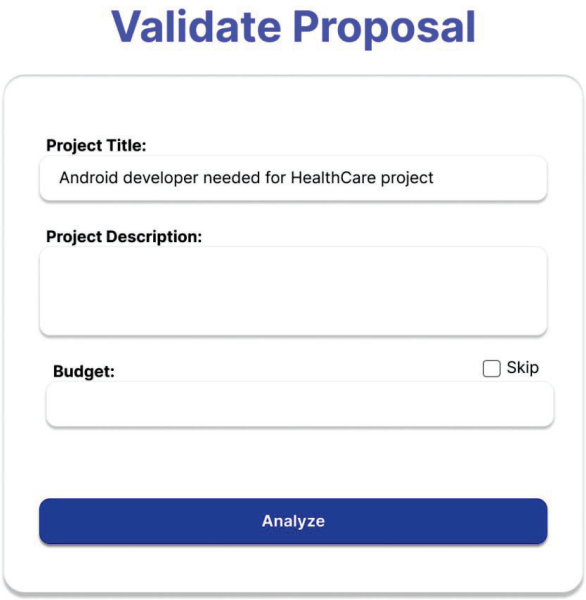

Рисунок 6.15 – Веб-сторінка заповнення основних даних щодо проекту

Після заповнення форми вона може виглядати наступним чином:

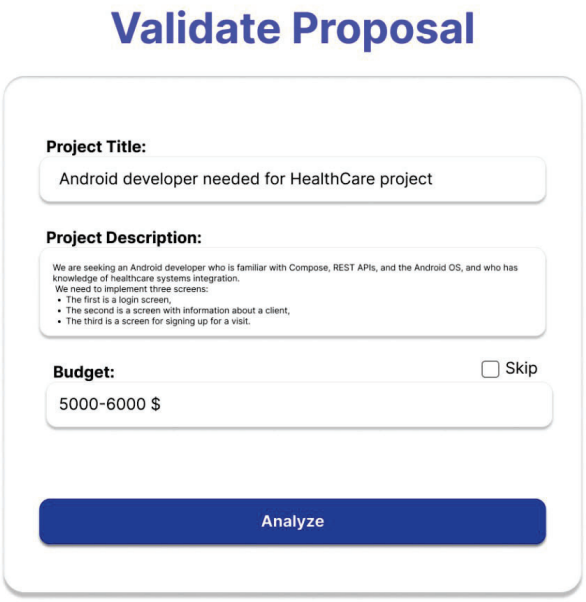

Рисунок 6.16 – Веб-сторінка з введеними даними щодо проекту

Крок 3: Перевірка доступності спеціалістів

На цьому етапі форма отримує результати GPT-аналізу поточного навантаження команди. Якщо немає вільних ресурсів у передбачені строки, з’являється візуальне сповіщення з вибором варіантів:

− надіслати запит на підбір нового фахівця (проект отримає статус "Pending");

− відмовитися від проекту (проект отримає статус "Rejected");

− у разі наявності доступних спеціалістів – користувач без додаткових дій переходить до наступного кроку.

Крок 4: Введення/перевірка термінів виконання

Користувач може ввести бажаний термін реалізації, або підтвердити той, що вказаний замовником. На основі попередніх кроків, GPT оцінює приблизний термін реалізації. Якщо термін виконання збігається або перевищує оцінку GPT – користувач переходить далі. У разі невідповідності з’являється повідомлення про неможливість дотримання строків із можливістю залишити коментар або запит на уточнення від замовника.

Крок 5: Автоматична естимація

На завершальному етапі формується короткий статистичний звіт із такими полями:

− загальна інформація про проект;

− аналіз вимог;

− оцінка складності реалізації;

− оцінка бюджету та часу;

− рекомендації;

− висновок.

Як результат опрацювання заявки можна отримати сформований ChatGPT документ (рисунок 6.17).

Звіт

з оцінки технічної реалізованості проекту «Android-додаток для охорони здоров'я»

1. Загальна інформація про проєкт

Назва: Android-додаток для HealthCare-платформи

Мета: Розробка мобільного застосунку з трьома основними екранами:
- Екран входу (Login Screen)
- Екран інформації про клієнта (Client Info Screen)
- Екран запису на прийом (Visit Signup Screen)

Технології:
- Мова розробки: Kotlin
- UI-фреймворк: Jetpack Compose
- Інтеграції: REST API
- Інші вимоги: Базове розуміння систем охорони здоров'я, можливе використання HL7/FHIR протоколів

Бюджет: 5000–6000 USD

2. Аналіз вимог

2.1 Ключові функціональні компоненти:

Компонент	Опис
Login Screen	Аутентифікація користувача, введення логіна/пароля, обробка токена
Client Info Screen	Отримання і вивід персональних даних з серверу
Visit Signup Screen	Вибір лікаря/часу, форма запису на прийом, обробка POST-запиту

Рисунок 6.17 – Приклад документу «Звіт з оцінки технічної реалізованності проєкту»

2.2 Нефункціональні вимоги:

- Висока продуктивність
- UX/UI відповідно до сучасних практик Material Design
- Обробка помилок при роботі з API
- Базове тестування (unit/UI tests)
- Можливість масштабування проєкту

3. Оцінка складності реалізації

Кількість екранів: 3 – невелика кількість
Технічна складність: Середня (Jetpack Compose + REST API)
Інтеграція з healthcare-системами: Може бути складною при використанні FHIR/HL7 – залежить від специфікації API
Доступність API: Якщо API існує та задокументований – реалізація значно спрощується

Висновок: Реалізація проєкту цілком можлива у вказаному бюджеті, за умови наявності backend API та базових специфікацій.

4. Оцінка бюджету та часу

4.1 Таблиця оцінки вартості

Етап	Опис	Орієнтовні години	Орієнт. ставка ($/год)	Вартість ($)
Login Screen	UI + інтеграція з API, валідація, обробка токена	20	50	1000
Client Info Screen	UI + REST запити, відображення персональних даних	25	50	1250
Visit Signup Screen	UI + логіка обробки форм, валідація, надсилання на	30	50	1500

Рисунок 6.17, аркуш 2

	сервер			
Загальна інтеграція API	Обробка запитів, помилок, сесій, авторизація	15	50	750
Тестування та дебаг	Unit/UI тести, виправлення помилок, перевірка стабільності	10	50	500
Фінальні роботи	Збірка APK, CI/CD (якщо потрібно), документація	10	50	500

Підсумок:
- Загальна тривалість: ≈ 110 годин
- Загальна вартість: $5500
- Тривалість розробки: 3–4 тижні (1 Android-розробник, повний робочий тиждень)

5. Рекомендації

1. Уточнити специфікацію API, особливо якщо є вимога до FHIR/HL7.
2. Наявність UI/UX дизайну значно пришвидшить розробку – бажано надати готові макети.
3. Якщо бекенд ще не реалізований, варто врахувати додаткові витрати (від $3000).
4. План на майбутнє: реалізація додаткових функцій (нагадування, чат, карта лікарень тощо) — можливе розширення MVP.

6. Висновок

Розробка Android-додатку для системи охорони здоров'я із трьома базовими екранами є реалістичною у межах бюджету 5000–6000 USD. Проєкт доцільно реалізовувати поетапно, починаючи з MVP, з подальшою можливістю масштабування.

Дата проведення оцінки: 06.05.2024 **Лідогенератор: Іванов А. Б.**

Рисунок 6.17, аркуш 3

Формат поданого документа, створеного на основі заявки, є узагальненою експертною оцінкою майбутнього проєкту, що забезпечує прозорість та обґрунтованість ухвалених рішень на етапі попереднього

відбору. У наведеному прикладі йдеться про оцінку ІТ-проєкту в галузі охорони здоров'я, що передбачає розробку Android-додатку для лікарів та пацієнтів. Документ містить низку ключових компонентів, які свідчать про ретельно продуману методику формування висновків. Насамперед привертає увагу чітко визначена мета та контекст використання програмного забезпечення – автоматизація медичних процесів та покращення комунікації між лікарем і пацієнтом. Це одразу дозволяє фахівцям із попереднього відбору сформувати уявлення про специфіку продукту та його ринкову нішу.

Оцінка функціональних вимог у документі має не лише описовий, але й прогностичний характер. У тексті надається орієнтовний перелік модулів та функцій, які мають бути реалізовані: від управління особистими кабінетами до відображення медичних записів та результатів обстежень. Такий підхід дозволяє краще зрозуміти складність задачі, рівень залучення ІТ-фахівців, а також можливі технічні ризики. Пояснення супроводжуються інформацією щодо технологій, які планується використовувати для реалізації – зокрема, згадуються Android SDK, Kotlin та інтеграція з зовнішніми базами даних через API. Це забезпечує прозорість для майбутніх розробників та замовників, полегшуючи оцінку відповідності заявки технічним вимогам компанії.

Окрему увагу в документі приділено часовим та фінансовим параметрам проєкту. Детально прописано розподіл задач за етапами реалізації: від початкового аналізу та проєктування до етапу тестування й запуску в експлуатацію. Такий поетапний підхід не лише полегшує планування, але й дозволяє аналітику чи менеджеру проєктів визначити вузькі місця або потенційно критичні точки реалізації. Вартісна оцінка кожного етапу дозволяє одразу сформувати уявлення про бюджет проєкту та зіставити його з внутрішніми можливостями ІТ-компанії, що відбирає заявки. Це – ключовий момент для ефективного прийняття рішень, адже в реальному бізнес-середовищі саме бюджетні обмеження часто визначають

доцільність реалізації певної пропозиції.

Документ також включає висновки щодо командної структури проєкту, зазначаючи кількість необхідних спеціалістів, їхні ролі та компетенції. Це має велике значення, адже дозволяє оцінити реалістичність ресурсного планування. Пропозиція, що потребує умовно десяти фахівців, серед яких кілька профільних інженерів, UI/UX дизайнер, аналітик і тестувальник, буде автоматично порівнюватися із поточним кадровим потенціалом компанії. Таким чином, документ дозволяє не лише технічно оцінити заявку, а й зробити управлінську експертизу, що виходить за межі суто розробницьких аспектів.

Таким чином, документ, сформований після опрацювання заявки, є не лише інформативним, а й функціонально насиченим. Він виконує роль універсального носія інформації, яка охоплює як технічні аспекти розробки, так і бізнес-параметри, пов’язані з ефективністю реалізації. Саме завдяки такому підходу система попереднього відбору отримує інструмент, що сприяє ухваленню рішень на основі даних, а не припущень, і зменшує ймовірність помилкової інвестиції в ненадійні чи неякісно підготовлені проєкти.

Користувачу доступно декілька опцій після опрацювання заявки. Він може викачати сформований звіт як документ, роздрукувати звіт на принтері або ж перейти на оформлення звіту для іншого проекту. У разі якщо користувач обрав варіант викачати звіт, то він має можливість перегляду сформованого висновоку, після чого передає результати технічному спеціалісту для фінального затвердження (рисунок 6.18).

Рисунок 6.18 – Веб-сторінка завершення аналізу

Отже, візуальний інтерфейс "Proposal Analyzer" реалізований відповідно до вимог сучасного UX-дизайну. Кожен його елемент має чітко визначену функцію, що відповідає послідовному логічному сценарію використання. Покрокова форма сприяє систематизованому збору інформації, а інтеграція GPT-підказок дозволяє зменшити навантаження на користувача при прийнятті рішень.

Завдяки такій структурі інтерфейсу, користувачі швидко адаптуються до сервісу, а кожен крок процесу – від логіну до верифікації проекту – проходить безперервно, зрозуміло й ефективно. Цей інтерфейс слугує не лише засобом введення даних, а й інструментом аналітики й контролю якості попереднього відбору IT-пропозицій.

7 ОБҐРУНТУВАННЯ ЗАСОБІВ ЗАХИСТУ ІНФОРМАЦІЇ ВІД НЕСАНКЦІОНОВАНОГО ДОСТУПУ

7.1 Реалізація допоміжного функціоналу

У рамках реалізації сервісу для автоматизації валідації та оцінки IT-проектів із використанням ChatGPT як асистента, важливу роль відіграють допоміжні функції, що забезпечують безпечний доступ, ідентифікацію та контроль доступу користувачів. До таких функцій належать: реєстрація з перевіркою корпоративного домену, авторизація користувача та обробка ролей, що визначають рівень доступу до функціоналу сервісу. Додатково реалізовано API-інтерфейс, який дозволяє взаємодіяти з серверною частиною через запити HTTP, а також використовується сучасна архітектура з Node.js, Express, PostgreSQL, bcrypt та JWT.

7.2.1 Структура допоміжного функціоналу

У межах створення сервісу відбору пропозицій на розробку IT-проектів важливу роль відіграє не лише основна бізнес-логіка сервісу, а й так званий допоміжний функціонал, який забезпечує надійність, безпеку та гнучкість роботи всіх користувачів. Така функціональність формує технічне підґрунтя для належної взаємодії між системою та користувачем, а також підтримує контроль доступу, обробку сесій та захист критично важливої інформації.

Допоміжний функціонал у цьому сервісі охоплює кілька ключових напрямів: процедури реєстрації з валідацією корпоративного домену, процес авторизації з використанням сучасних методів шифрування, реалізацію ролей для розмежування прав доступу, а також загальну систему

захисту даних. Всі ці компоненти не є безпосередньо пов’язаними з основною логікою підбору проектів, однак вони створюють критичну інфраструктуру, без якої сервіс не зміг би ефективно функціонувати в умовах реального бізнес-середовища (рисунок 7.1).

Рисунок 7.1 – Use Case діаграма зображення допоміжних сервісів

Реалізація реєстрації з перевіркою доменного імені дозволяє обмежити використання платформи лише уповноваженими представниками компаній, виключаючи випадкових або сторонніх користувачів. Це, в свою чергу, покращує загальну безпеку та контрольованість доступу. Система авторизації, заснована на протоколі JWT, забезпечує безперервність сесії користувача та дозволяє здійснювати запити до сервісу без необхідності повторного входу, що покращує зручність користування системою.

Наступним важливим компонентом є обробка ролей користувачів, яка формує ієрархію доступу до різних функцій сервісу. Такий підхід дозволяє призначати користувачам лише ті функції, які відповідають їх посадовим обов’язкам, тим самим запобігаючи випадковому або навмисному доступу до конфіденційної чи технічної інформації.

Крім того, безпека даних залишається пріоритетом під час розробки сервісу. Усі критичні операції супроводжуються багаторівневою перевіркою, а паролі користувачів зберігаються у вигляді хешів, створених за допомогою стійкого криптографічного алгоритму bcrypt. Аутентифікаційні токени мають обмежений термін дії, що дозволяє зменшити ризики скомпрометованих сесій.

Загалом, структура допоміжного функціоналу виконує роль опорного каркасу, який дозволяє основній системі діяти ефективно, безпечно та масштабовано. У наступних підрозділах буде детально розглянуто реалізацію кожного з цих компонентів, включаючи алгоритми роботи, принципи перевірки та впроваджені механізми контролю доступу.

7.2.1.1 Реєстрація користувача з перевіркою домену

Реєстрація – це процес створення нового облікового запису користувача з валідацією email-адреси за доменом. В рамках даного сервісу дозволяється реєстрація лише з корпоративних поштових доменів. Такий підхід дозволяє обмежити доступ до сервісу лише для уповноважених співробітників або представників компанії.

Алгоритм дій:

− користувач вводить ім’я користувача (username), email, пароль та роль у форму;

− сервер перевіряє, чи входить домен email до списку дозволених (наприклад, company.com, corp.io);

− у разі відповідності, пароль хешується за допомогою bcrypt і створюється запис у базі даних users.

Якщо домен недозволений – виводиться повідомлення про помилку (рисунок 7.2).

```
const allowedDomains = ['company.com', 'corp.io'];
const emailDomain = email.split('@')[1];
if (!allowedDomains.includes(emailDomain)) {
  throw new Error('Реєстрація дозволена лише для корпоративних адрес');
}
```

Рисунок 7.2 – Обробка валідації реєстрації по домену користувача

7.2.1.2 Авторизація (логін)

Це процес ідентифікації користувача за email та паролем. Після успішного входу видається токен доступу (JWT), який зберігається на клієнтській стороні та використовується для подальших запитів до сервісу (рисунок 7.3).

Етапи авторизації:

- користувач надсилає email та пароль на сервер;
- сервер знаходить користувача в базі за email;
- порівнюється введений пароль із хешованим у базі;
- якщо все правильно – генерується JWT-токен з ідентифікатором та роллю;
- токен повертається клієнту для збереження.

```
const user = await getUserByEmail(email);
if (!user || !comparePasswords(inputPassword, user.password_hash)) {
    throw new Error('Невірна електронна адреса або пароль');
}
```

Рисунок 7.3 – Реалізація авторизації користувача

7.2.2 Обробка ролей

У полі role таблиці users задається роль, яка визначає доступ користувача до тих чи інших функцій:

- sales – подання запитів на оцінку;
- developer – технічне редагування проектів;
- leadgen – первинне завантаження проектів.

Ролі дозволяють реалізувати гнучке управління правами доступу.

7.2.3 Захист інформації

Для забезпечення безпеки користувацьких паролів застосовується алгоритм хешування bcrypt, який гарантує надійний захист від несанкціонованого доступу. Токени автентифікації мають строго обмежений термін дії, наприклад, вони дійсні лише протягом двох годин, що знижує ризик компрометації сесій. Крім того, всі дані проходять ретельну перевірку та валідацію на кожному етапі обробки, що дозволяє запобігати можливим помилкам або атакам, пов’язаним із некоректним введенням інформації.

8 ТЕСТУВАННЯ ТА ОЦІНКА НАДІЙНОСТІ ФУНКЦІОНУВАННЯ ПРОГРАМНИХ І ТЕХНІЧНИХ РІШЕНЬ

8.1 Тестування програмного рішення

Тестування програмного забезпечення як етап забезпечення якості користувацьких додатків

Проведення тестування програмного забезпечення є критично важливим етапом у процесі створення будь-якого прикладного сервісу, орієнтованої на кінцевого користувача [6]. Саме завдяки тестуванню з’являється можливість виявити дефекти, збої або помилки у роботі сервісу ще до моменту її розгортання у реальному середовищі. Така перевірка дозволяє усунути недоліки та зробити програму більш стабільною, зручною та безпечною для експлуатації. Тестування виступає обов’язковим компонентом перед впровадженням готового продукту, оскільки забезпечує можливість своєчасного виявлення критичних помилок, які потенційно можуть призвести до негативного досвіду користувача або до втрати даних.

У рамках контрольного аналізу функціонування сервісу здійснено кілька видів тестувань, кожен із яких мав на меті перевірити окремий аспект ефективності та стабільності програмного забезпечення. Основними напрямками проведеного тестування стали:

- перевірка стійкості до навантаження;
- оцінка надійності роботи під інтенсивним використанням;
- аналіз зручності користування графічним інтерфейсом;
- аналіз результатів тестування.

У результаті проведеного навантажувального тестування було встановлено, що сервіс у поточній конфігурації повністю справляється з обсягом роботи, передбаченим базовими технічними вимогами. Під час імітації середнього навантаження (близько 500 одночасних запитів) середнє завантаження центрального процесора не перевищувало 65%, а

використання оперативної пам’яті стабільно трималося в межах 70–75% доступного обсягу. При цьому середній час відповіді API не перевищував 320 мс. Однак при подальшому розширенні користувацької бази (понад 1500 одночасних сесій) спостерігалося зростання завантаження процесора до 90–92%, що потенційно може призвести до затримок у відповіді сервісу. Для забезпечення стабільної роботи у випадку масштабування доцільним буде перехід на сервери з більшими обчислювальними ресурсами, зокрема процесорами з більшою кількістю ядер та збільшеним обсягом оперативної пам’яті, наприклад до 32 ГБ.

Оцінка надійності показала, що сервіс демонструє стабільну роботу навіть у разі підвищених навантажень. У процесі тестування було зафіксовано кілька короткочасних піків використання оперативної пам’яті серверної частини до 88–90%, що частково впливало на швидкодію – у ці моменти спостерігалося збільшення часу відповіді до 700–800 мс. Причиною цього є обмежений поточний обсяг пам’яті (16 ГБ) для роботи з великим обсягом одночасних запитів. Вирішенням даної проблеми є технічна модернізація серверного середовища – збільшення обсягу оперативної пам’яті до 32 ГБ та оптимізація внутрішніх алгоритмів кешування.

Тестування інтерфейсу користувача засвідчило, що сервіс є зручним та зрозумілим в навігації. Користувачі легко орієнтуються у функціональних елементах: кнопках, меню, посиланнях, а також мають змогу безперешкодно взаємодіяти з усіма необхідними модулями. Жодних порушень у роботі клієнтської частини сайту не виявлено – як з боку маршрутизації, так і в аспекті функціональності елементів керування. Учасники тестування також підтвердили, що інформаційне наповнення інтерфейсу повністю відповідає їхнім очікуванням та потребам, надаючи всі необхідні відомості для зручної взаємодії з платформою.

За підсумками проведених тестів можна впевнено стверджувати, що розроблене програмне забезпечення відповідає як функціональним, так і

візуальним вимогам до сучасних інформаційних систем. Сервіс продемонстрував стабільність роботи, зручність використання, адаптивність до майбутнього масштабування та технічну надійність. Це дозволяє рекомендувати його до впровадження в реальне середовище експлуатації з урахуванням перспективи подальшого вдосконалення серверної інфраструктури.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи розроблено IT-сервіс, що автоматизує процес відбору пропозицій на розробку IT-проектів IT-компанії. Запропонований сервіс забезпечує ефективну взаємодію між замовниками проектів і командами розробників, враховуючи відповідність заявок встановленим критеріям відбору, пріоритетам та технічним параметрам.

Архітектура програмного продукту побудована із використанням сучасного технологічного стеку: на клієнтській частині застосовано бібліотеку React.js для створення динамічного та інтуїтивно зрозумілого інтерфейсу користувача, на серверній частині – Node.js з фреймворком Express для розробки серверної логіки, а також PostgreSQL як основну систему керування базами даних, що забезпечує гнучке управління інформаційними моделями та ефективне збереження даних.

У процесі реалізації проекту виконано низку ключових задач:

− проведено глибокий аналіз предметної області, пов’язаної з автоматизацією процесу відбору проектних пропозицій;

− спроектовано логічну структуру сервісу з урахуванням ролей користувачів, етапів відбору та критеріїв оцінювання;

− розроблено концепцію бази даних з урахуванням усіх сутностей і зв’язків, включаючи заявки, проекти, критерії, теги та користувачів;

− реалізовано серверну частину з авторизацією, обробкою даних та захистом запитів через JWT;

− розроблено користувацький інтерфейс із дотриманням принципів UX/UI-дизайну для полегшення доступу до основних функцій.

Проведено тестування стабільності сервісу, його продуктивності та зручності користування для підтвердження якості розробленого рішення.

Таким чином, реалізовано стабільний IT-сервіс, який задовольняє

основні потреби компанії в автоматизованому відборі проектів, сприяє зменшенню витрат часу на прийняття рішень та забезпечує прозору процедуру взаємодії між учасниками процесу.

У подальшому, розвиток сервісу передбачає розширення його функціоналу. Зокрема, доцільним є впровадження модуля аналітики для відстеження ефективності поданих заявок, а також додавання модуля формування звіту на основі машинного навчання для автоматичного добору проектів відповідно до профілю виконавця. Крім того, перспективним напрямом є створення мобільної версії платформи з можливістю пуш-сповіщень, що забезпечить більшу доступність сервісу та інтеграцію в робочі процеси користувачів у режимі реального часу.

Це свідчить про потенціал розробленого сервісу до масштабування, вдосконалення та подальшого використання в умовах динамічного середовища IT-індустрії.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Методичні вказівки до передатестаційної практики (для студентів спеціальності 122 «Комп'ютерні науки та інформаційні технології») / Упоряд.: Доля О.Є. - Харків: ХНУРЕ, 2021.

2. Про університет. URL: https://nure.ua/universytet/pro-universitet (дата звернення 12.03.2025).

3. Положення про організацію освітнього процесу в Харківську національну університеті радіоелектроніки (затверджено наказом ХНУРЕ від 27.11.2020 №400). Харків: ХНУРЕ, 2020. URL: https://nure.ua/wp-content/uploads/Main_Docs_NURE/polozhennja-pro-organizaciju-osvitnogo-procesu-v-hnure.pdf (дата звернення: 14.03.2025).

4. Статут Харківського національного університету радіоелектроніки (нова редакція), погоджений Конференцією трудового колективу 26 грудня 2017 року. URL:https://nure.ua/wp content/uploads/Main_Docs_NURE/statut.pdf (дата звернення: 14.03.2025).

5. Кафедра Інформаційних управляючих систем Харківського національного університету радіоелектроніки. URL: https://nure.ua/ru/department/kafedra-ius (дата звернення: 15.03.2025).

6. Project Proposals and the Project Selection Process. URL: https://www.ittoolkit.com/articles/project-proposals (дата звернення: 22.04.2025).

7. Project Selection Methods for Project Managers. URL: https://www.simplilearn.com/project-selection-methods-article (дата звернення: 22.04.2025).

8. Project Selection: Use These 8 Selection Methods for Better Results. URL: https://www.projectmanager.com/blog/project-selection-for-better-strategic-results (дата звернення: 22.04.2025).

9. How to Write a Software Proposal (With Examples). URL:

https://www.proposify.com/blog/how-to-write-a-software-proposal (дата звернення: 22.04.2025).

10. Elements Every Project Proposal Should Include. URL: https://www.kantata.com/blog/article/5-elements-every-project-proposal-should-include (дата звернення: 22.04.2025).

11. AI-driven RFP tools for proposal managers. URL: https://www.responsive.io/blog/how-to-choose-a-proposal-management-solution (дата звернення: 22.04.2025).

12. Малькова І.А., Маланія І.Д. Оптимізація відбору пропозицій на розробку ІТ-проєктів з використанням штучного інтелекту. Modern Perspectives on Science and Economic Progress. Collection of Scientific Papers with Proceedings of the 1st International Scientific and Practical Conference. International Scientific Unity. June 4-6, 2025. Vilnius, Lithuania. 136-139 p. URL: https://isu-conference.com/en/archive/modern-perspectives-on-science-and-economic-progress-4-06-25/ https://doi.org/10.70286/isu-04.06.2025

13. Шеховцова, В. І., Малькова, І. А., Потапенко, А. О., & Клименко, Д. А. (2024). Інформаційна технологія обґрунтування та формування ціннісної пропозиції. *АСУ та прилади автоматики*, *1*(183), 46–61. DOI: https://doi.org/10.30837/0135-1710.2024.183.046